Joint position estimation, packet routing and sleep scheduling in Wireless Sensor Networks

by

Maurício Bertanha

A thesis submitted in partial fulfillment
of the requirements for the degree of

Masters of Science

in

Computer Science

Faculty of Business and Information Technology
University of Ontario Institute of Technology

Supervisor: Dr. Richard W. Pazzi

August 2017

Abstract

Wireless Sensor Network (WSN) is an important research field in Computer Science with applications that span multiple domains. Due to the limitation of sensor nodes, network lifetime is a critical issue that needs to be addressed. Therefore, in this thesis I propose the Energy-aware Connected k-Neighbourhood (ECKN), a joint position estimation, packet routing, and sleep scheduling solution that combines some overlapping features. I propose a localization algorithm that performs trilateration using the position of a mobile sink and of neighbour nodes to estimate the position of a sensor node with no GPS module. I introduce a routing protocol based on the well-known Greedy Geographic Forwarding (GGF). Similarly to GGF, my protocol takes into consideration the position of neighbours to decide the best forwarding node, however it also considers the residual energy in order to guarantee that the forwarding node will deliver the packet. The concept of bridges is also introduced, in which the sink compares its current position with previous positions and calculates whether there is a shortest path in order to create a bridge that will reduce the number of hops a packet has to travel through. Lastly, a sleep scheduler is proposed in order to extend the network lifetime, it is based on the Connected k-Neighbourhood (CKN) algorithm, which aids in the decision of what nodes goes to sleep while maintaining the network connected. My sleep scheduler maintains the network denser in the area close to the sink, since this region receives packets from the whole network to forward to the sink.

An extensive set of performance evaluation experiments is conducted and results show that ECKN can extend network lifetime, while sustaining acceptable packet delivery ratio and reducing network overhead.

Acknowledgements

I am sincerely grateful to Dr. Richard W. Pazzi for his expert advice, guidance and encouragement during my master's studies.

I would like to offer special gratitude to my family and friends, specially my parents, my siblings, and my girlfriend for their unconditional love, constant support, and encouragement.

I would also like to thank my friends in the Advanced Networking Technology and Security (ANTS) Lab.

Lastly, the financial support of the Natural Sciences and Engineering Research Council of Canada (NSERC) is gratefully acknowledged.

Publications

The following publications resulted from the work in this thesis:

Mauricio Bertanha and Richard Pazzi, 2016. LOGR: Joint LOcalization and Geographic Routing-based Data Dissemination in Wireless Sensor Networks with Mobile Sinks. In *Proc. of the 6th ACM Symp. on Development and Analysis of Intelligent Vehicular Networks and Applications* (pp. 83-90). ACM.

Mauricio Bertanha and Richard Pazzi, 2017. JLPR: Joint Range-Based Localization Using Trilateration and Packet Routing in Wireless Sensor Networks with Mobile Sinks. In *IEEE Symp. on Computers and Comm. (ISCC)* (pp. 646–651). IEEE.

Mauricio Bertanha and Richard Pazzi, 2017. ECKN: Joint position estimation, packet routing and sleep scheduling in Wireless Sensor Networks. Submitted to *Ad Hoc Networks* (pp. To appear).

[Poster] Mauricio Bertanha, Tomo Nikolovski and Richard Pazzi. 2016. Geographic Sleep Scheduling in Duty-Cycled Wireless Sensor Networks. In *Proc. of the 6th Developing Next Generation Intelligent Vehicular Network and Applications (DIVA) Workshop*. ACM.

[Poster] Tomo Nikolovski, Mauricio Bertanha and Richard Pazzi. 2016. Heterogeneous (DSRC and LTE) Event-Based Data Dissemination Protocol for VANETs. In *Proc. of the 6th Developing Next Generation Intelligent Vehicular Network and Applications (DIVA) Workshop*. ACM.

Contents

List of Figures

List of Tables

List of Algorithms

Abbreviations

AOA Angle of Arrival.

ASCENT Adaptive Self-Configuring sEnsor Networks Topologies.

CKN Connected k-Neighbourhood.

ECKN Energy-aware Connected k-Neighbourhood.

EZBLM Enhanced Zone-Based Localization Method.

FACE-2 Face Traversal Technique.

GG Gabriel Graph.

GGF Greedy Geographic Forwarding.

GPS Global Positioning System.

GPSR Greedy Perimeter Stateless Routing.

MAC Media Access Control.

MPR Multi-Point Relay.

OLSR Optimized Link State Routing.

PTW Pipelined Tone Wakeup.

RND Regulated Neighbourhood Distance.

RNG Relative Neighbourhood Graph.

ROULA OLSR based Localization.

RSSI Received Signal Strength Indicator.

STEM Sparse Topology and Energy Management.

TTL Time to Live.

WSN Wireless Sensor Network.

ZBLM Zone-Based Localization Method.

Chapter 1

Introduction

A Wireless Sensor Network (WSN) [2, 7, 50] consists of a large number of low-cost devices, or sensor nodes, with wireless communication and it enables several applications such as in health, military, and security [3]. In this type of network, the autonomous sensor nodes are used to track, monitor or control a large area without the need of fixed infrastructure [10, 21, 23, 52], which facilitates installation and maintenance of the network.

The general idea of a WSN is that each sensor node, after gathering some information to be analyzed, sends its data until it reaches a sink, where this information can be processed. Due to the limited radio range, the data has to be routed through other nodes in an ad hoc way. Communication is made through wireless protocols ZigBee/IEEE 802.15.4 [18,53] or IEEE 802.11 (Wi-Fi) [18], which offer an implementation for the lower layers, Physical and Media Access Control (MAC), for a typical WSN.

Coverage and connectivity are two performance metrics mainly studied in WSNs according to Li et al. [35]. Coverage refers to the surveillance map of the area. In this metric, nodes are placed in the region and cooperate among them in order to maintain

a good quality of coverage. Some benchmarks are used in terms of granularity to quantify coverage quality. In the highest granularity every point of the map should be covered by at least one sensor. In the medium granularity every path crossing the network should be covered. Finally, in the lowest granularity the map is not entirely covered by the nodes and the network is not necessarily connected. Connectivity refers to message retrieval and delivery in the network. The connectivity of a node to other nodes of the network can be considered from two characteristics regarding energy consumption. The temporal domain refers to nodes switching between active and sleep states. The spatial domain refers to nodes having multiple energy levels, varying its communication range.

1.1 Problem Statement

In WSN applications, location-awareness plays an important role to fundamental tasks such as packet routing, event mapping, and energy savings. Due to the deployment of a large number of sensor nodes, manual placement of these devices is often unfeasible and costly. Consequently, sensor nodes usually acquire their positions through location estimation. One of the most common localization method is the Global Positioning System (GPS) [29]. However, for small, inexpensive, low-power devices that are left unattended for long periods of time, the use of GPS modules on all sensor nodes is unfeasible due to size, form factor, cost, and power constraints [13].

In my publications [8,9], I proposed a protocol that estimates sensor node positions by exchanging position packets between a mobile sink and the sensor nodes. Since the sink is more powerful, it is able to house a GPS module and share its position within the network. Once a sensor node receives a position packet, it estimates its own position using a simple trilateration algorithm using the position and the Received

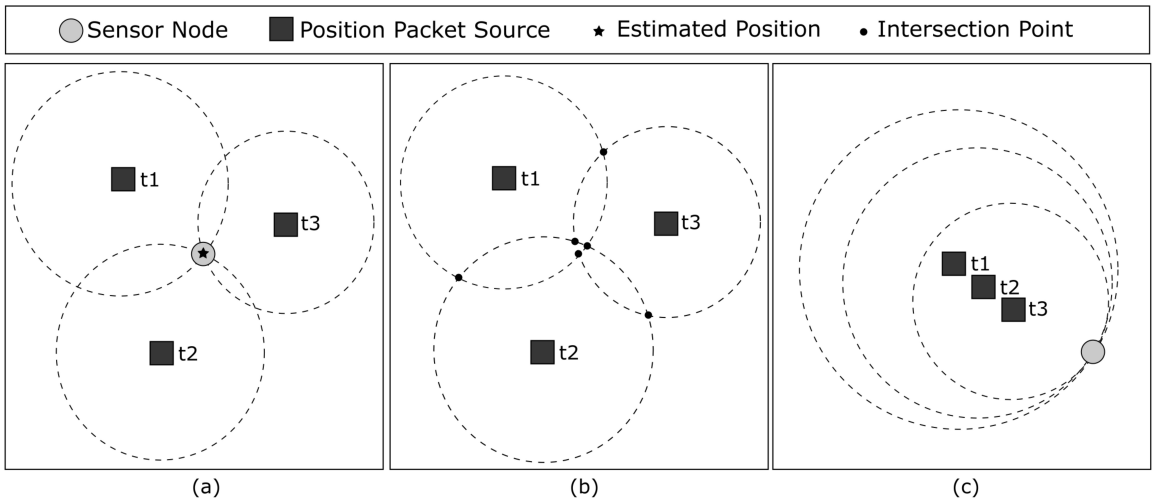

Figure 1.1: (a) Position of sensor node defined by Trilateration. (b) GPS error added to sink packets, intersection between the packets is not a single point. (c) Poor position packet samples to perform trilateration.

Signal Strength Indicator (RSSI) [12]. A qualified node is calculated with at least three position packets, resulting in a more accurate position estimation, as can be seen in Figure 1.1(a). However, with only one or two samples it is still possible to have a rough estimate. With only one sample, the sensor node assumes that it is at the same position as the sink. With two samples, it considers its position as the midpoint between the intersection of the position packets circles.

Although trilateration is a viable solution for the localization problem, there are some issues that need to be addressed. GPS location is not 100% accurate since noise creates some uncertainties in its estimations [47]. When receiving position packets and calculating the intersection between them, the sensor node is not able to find a common point where they intersect. Instead, it ends up with six different intersection points, as can be seen in Figure 1.1(b). Using this information, the sensor node has to estimate its position by deciding which intersection points, called reference points, are the best to be used in the estimation. Furthermore, considering the worst case scenario where the sink moves randomly within the WSN boundary, it implies

that some position packets might not be ideal to perform trilateration. Figure 1.1(c) shows an example of poor samples, in which collinear position packets are received by a sensor node. This makes it impossible for the trilateration algorithm to estimate an accurate position.

In a WSN, communication between source and destination nodes may require a multi-hop communication strategy since the radio range and other resources are constrained. As previously mentioned, by being aware of their position, nodes can properly forward the messages to their destination. This information is widely exploited to improve the performance of routing protocols. A popular approach to this routing issue is Geographic Routing.

Geographic Routing is one of the most popular routing schemes in WSNs, mainly because it scales better: the network size is not a problem in this scheme. The forwarding decision is taken based on the position of the node and its neighbours. The most straightforward algorithm to address geographic routing is Greedy Geographic Forwarding (GGF) [31]. The idea is to forward the message to the neighbour node that is closest to the destination. Figure 1.2 represents one example of Greedy Geographic Forwarding. Here, Node 1 receives the message destined to the sink and then forwards this message to Node 3, which is the closest neighbour to the destination. If there is no neighbour closer to destination, GGF fails and the message is dropped.

A number of recovery strategies for GGF failure have been proposed in the literature. One of the most popular approaches is face routing, first proposed by Kranakis et al. [33]. The network is divided into empty zones, i.e., faces, surrounded by the nodes that are inter-connected. Face routing requires a planar network graph, i.e., it cannot have any crossing links. Bose et al. [11] explains the network graph planarization algorithm, which computes the intersection of node neighbours with a well-known planar graph. After obtaining the faces, it is possible to use this information to calcu-

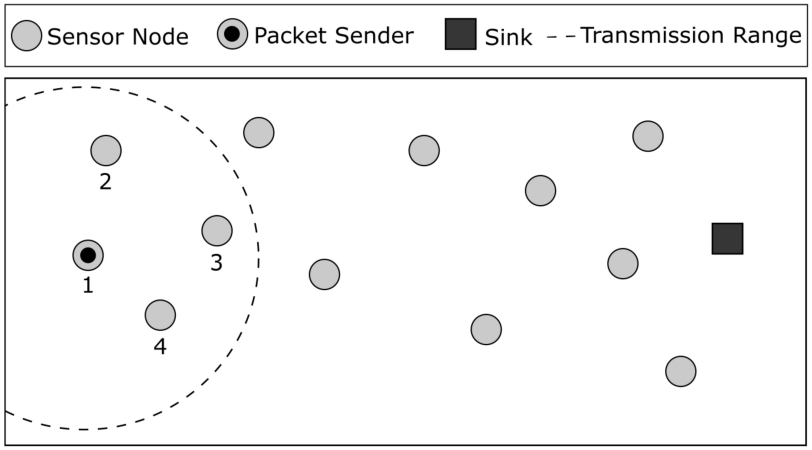

Figure 1.2: Greedy Geographic Forwarding Example. Node 3 is going to receive the packet because it is closer to the sink.

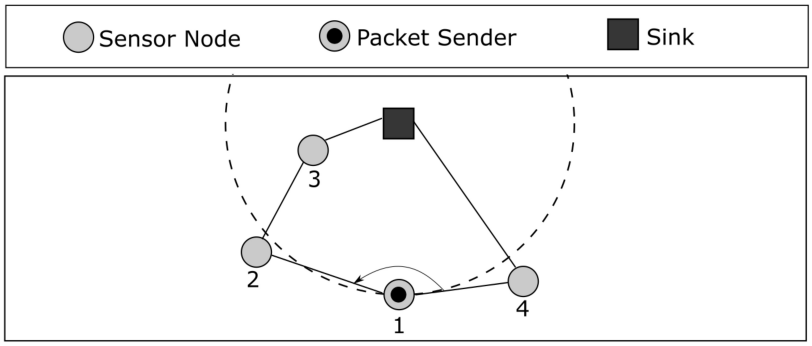

Figure 1.3: GGF fails because sensor node 1 is closer to the sink than its neighbours. A face is created with the nodes 1, 2, 3, 4 and the sink, following the right-hand rule, 1 sends the message to node 2, lying in clockwise direction.

late a path to the destination node by exploiting the faces by applying the right-hand rule. The right-hand rule sends the message to the edge lying in clockwise direction. The right-hand rule can be seen in Figure 1.3. This process is repeated until it reaches a node that is closer to the destination node, thus GGF is resumed from this node.

Recently, research efforts focus on duty-cycled approaches [37, 44, 52] in order to improve network lifetime while keeping a good delivery guarantee. Guaranteed delivery refers to the success ratio of forwarding a message from source to destination. It requires at least one path connecting these two nodes in the network. Existing

works on duty-cycle aim to achieve network coverage and/or connectivity. To this end, a duty-cycle algorithm has to find what are the best nodes to sleep in order to minimize the number of awake nodes while maintaining the network coverage and/or connectivity.

1.2 Thesis Contribution

This thesis proposes Energy-aware Connected k-Neighbourhood (ECKN), a joint approach for position estimation, packet routing and sleep scheduling in WSN. I noticed that this process shares some common features. My approach aims at using the sensor nodes position to perform the packet routing and to estimate what nodes are going to sleep. At the same time, the sink has to broadcast its position in order to perform the packet routing. Nodes can take advantage of this information to estimate their position. Furthermore, the sleep schedule algorithm intention is to keep the network connected, facilitating the packet routing algorithm to achieve a good success delivery ratio. By the combination of these approaches, I aspire to increase the network lifetime by eliminating any overhead the network might have.

I propose the Distance and Closeness algorithms to aid in the decision of which intersection points a sensor node should use to estimate its position:

- The Distance algorithm compares the distance between intersection points and considers the ones with smallest distances as reference points.
- The Closeness algorithm checks every possible combination of intersection points and considers the combination that has the smallest distance among them as reference points.

I also propose Position Distance, Circle Limit and a Hybrid version of the algo-

rithms in order to decide which position packets to use in trilateration while avoiding scenarios as the one depicted in Figure 1.1(c):

- The Position Distance algorithm calculates the distance between two position packet sources. If the distance is greater than a threshold, the packet is accepted.
- The Circle Limit algorithm checks whether a position packet is inside the circle of another position packet, and it accepts only those outside the circle.
- A Hybrid algorithm is also presented. Results show that Position Distance converges to acceptable accuracy faster, however, Circle Limit converges to more accurate estimations. My Hybrid algorithm starts by running Position Distance and, after getting the initial position estimations, it switches to Circle Limit.

Packet Routing is performed according to the GGF algorithm, but instead of the closest node to the sink, the sender sensor node decides the next hop using a utility function which considers both the sensor node position and its remaining energy. I also introduce the concept of bridges, which essentially tries to find shortest paths between previous positions of the sink and the current position.

Lastly, I propose a sleep schedule algorithm based on Connected k-Neighbourhood (CKN) [37]. Contrasting the original approach, I have different network densities according to the distance to the sink. My algorithm keeps the network denser in the region close to the sink, since there are more packets in this area to be forwarded to the sink. On the other hand, the network is more sparse in the region far from the sink because this area has less packets to be forwarded.

1.3 Thesis Organization

This thesis is organized as follows: Chapter 1 provides a brief overview to WSN, discussing about open problems in this area and presenting the contribution of this thesis. Section 2 presents a comprehensive literature review of sensor node position estimation, packet routing, and duty-cycling. Chapter 3 describes my proposed position estimation algorithm, packet routing algorithm, and sleep scheduler algorithm. The performance criteria and experimental results obtained via simulation are presented in Chapter 4. Lastly, Chapter 5 concludes the thesis.

Chapter 2

Related Work

In this chapter, I discuss related research works proposed by the scientific community on sensor node position estimation, packet routing, and duty-cycling in the field of WSNs.

2.1 Sensor Node Position Estimation

Existing efforts in sensor node position estimation [5, 27, 34] can be organized into two categories: Range-based and Range-free. Range-based approaches assume that sensor nodes are able to measure the range or distance between the sensor nodes. Range-free uses nondeterministic attributes of the network, such as closeness and hop count, to perform position estimation. Range-free usually shows lower accuracy than range-based solutions.

2.1.1 Ad hoc Positioning System (APS) using Angle of Arrival

DV-Bearing and DV-Radial are two range-based algorithms proposed by Niculescu and Nath [38]. They use Angle of Arrival (AOA), allowing sensor nodes to derive position information between immediate neighbours. AOA refers to the capability of nodes to sense the direction from which a signal is received. DV-Bearing allows each node to get a bearing to a landmark, while DV-Radial allows a node to get a bearing and a radial to a landmark. Assuming that a node has this information, it will be able to compute its own orientation with respect to the referred landmark, and forward it further into the network. With no additional infrastructure, the methods provide absolute coordinates and orientation, working well for disconnected networks.

2.1.2 A Range Based Localization System in Multihop Wireless Sensor Networks: A Distributed Cooperative Approach

Panday and Varma [40] proposed a cooperative range-based localization system, where they modify a range-based algorithm in order to save energy by reducing collisions and retransmission of range packets. To do so, it synchronizes the neighbouring nodes. The sender node first broadcasts a location request packet, the broadcasting is controlled in time scale by a timer. Upon receiving the request, the reference nodes unicast their location. The requesting node stores the location data along with the distance to the anchors. With this information, the requesting node is able to compute its position using multilateration method.

2.1.3 Regulated Neighbourhood Distance (RND)

Wu et al. [48] proposed a range-free localization method based on Regulated Neighbourhood Distance (RND), which represent the relative Euclidean distance between two neighbouring nodes. RND avoids the hop-distance ambiguity problem by relating the proximity of two neighbours to their neighbour partitions. In the proposed RND-based localization method, called DV-RND, each pair of neighbour nodes compute the RND in between them, with this value, any pair of nodes in the network can compute the shortest RND path in between them using any shortest path algorithm.

2.1.4 Zone-Based Localization Method (ZBLM)

Chen et al. [16] proposed Zone-Based Localization Method (ZBLM) and Enhanced Zone-Based Localization Method (EZBLM), two range-free localization methods that utilizes only two anchor nodes, placed on the bottom-left and bottom-right corners of a square region of the WSN, and uses bilateration to estimate node position by counting the minimum number of hops to the anchor nodes. The process consists into computing the minimum hop counts by flooding, dividing the monitored region into zones, and assigning the coordinate of sensors in each zone by bilateration.

2.2 Packet Routing

The two main classes of Packet Routing for WSNs [1,4,26] are Proactive and Reactive. Proactive routing protocols provide fast responses to topology changes by maintaining routing information for all network nodes. On the other hand, reactive routing protocols provide routing information on demand without the need of maintaining routing tables.

2.2.1 Greedy Perimeter Stateless Routing (GPSR)

Greedy Perimeter Stateless Routing (GPSR) is a known non-reactive routing algorithm in the literature proposed by Karp et al. [31]. A node knows its neighbours position by receiving a beacon periodically. To minimize the cost of beaconing, GPSR piggbacks the local sending node's position in all data packets it forwards and each node receives a copy of all packets for all nodes within radio range. Knowing each neighbour position, GPSR can use the GGF algorithm to send packets to destination nodes. If a GGF failure occurs, GPSR will use a perimeter forwarding algorithm to recover by exploiting the faces of a planar graph of the network topology. In order to avoid crossing links, GPSR uses an algorithm to remove edges that are not part of the Relative Neighbourhood Graph (RNG) [46] and/or Gabriel Graph (GG) [24], two planar graphs long-known in the literature. Having the planar graph for the network, GPSR uses the right-hand rule for traversing the edges of a face until it gets to a node that is closer to the destination than the location where packet entered perimeter mode.

2.2.2 Face Traversal Technique (FACE-2)

Face Traversal Technique (FACE-2) proposed by Bose et al. [11] creates a planar graph based on the network topology and uses face routing to find a path between source and destination nodes. Essentially, the algorithm creates a line segment from the source to destination, finds a face with the source node on its boundary that intersects this line segment, and traverses this face until reaching an edge that intersects the line segment. When reaching this edge, one of the nodes of the edge becomes the new source node and the process repeats until it reaches the destination node.

2.2.3 Integrated Protocol for Optimized Link State Routing and Localization (OLSR-L)

Mineno et al. [36] proposed an integrated protocol for Optimized Link State Routing (OLSR) [17] and OLSR based Localization (ROULA) [45] called OLSR-L, which implements routing and localization simultaneously. It was observed that communication overhead doubled when running localization and routing separately. Moreover, OLSR and ROULA present overlapping functionalities that can be integrated in order to reduce this communication overhead. Proposed by Clausen et al. [17], OLSR is a proactive routing protocol which key concept is Multi-Point Relay (MPR), selected nodes that forward broadcast messages during the flooding process. OLSR algorithm selects MPR nodes by flooding each node 1-hop nodes list to their 1-hop nodes, the selection is made when the node has the same 2-hop nodes list. Then, MPR nodes are selected as relay nodes to make the routing table. ROULA is a range-free and anchor-free localization protocol proposed by Takenaka et al. [45], nodes running this protocol search for other nodes that are arranged into regular triangles in order to satisfy the requirement of independency from anchor. ROULA algorithm uses MPR nodes to choose the farthest 2-hop node, which is a candidate to be a vertex of a regular triangle. Thereafter, each node floods the network with packets that carry a farthest 2-hop node list. Then, nodes calculate relative local coordinates by matching regular triangles on the basis of this information. Therefore, localization and routing are performed simultaneously with OLSR providing ROULA's required neighbour node information by periodically holding and updating 1-hop neighbour information. Moreover, ROULA uses OLSR's MPR node to estimate node distance accurately.

2.2.4 Simultaneous Localization and Routing in Sensor Networks using Shadow Edges

The solution proposed by Oliva et al. [39] addresses a multi-step simultaneous routing and localization procedure. During the clustering and routing step, nodes are collected into clusters. Clustering is performed according to CBRP [30] algorithm in order to retrieve single-hop clusters allowing both robust communication and localization. Exploiting the traffic generated by this step, kernel nodes send their position to neighbours, which are able to compute their position by trilateration and Shadow Edges. The key idea of Shadow Edges is to use the lack of information about the connectivity between nodes. Such an edge can be created when a node v_i has 2 localized neighbours, which results 2 localization options. One of them is discarded based on a fourth node v_j in the area, if the node v_i was in the localization option such that v_j lies in the corresponding disk, it should have sensed node v_j. Therefore node v_i belongs to the other localization option. During cluster localization step, the Shadow Edges localization procedure is performed. Only clusters having more than 3 non collinear, connected localized nodes are considered during this step and Shadow Edges inside the cluster are performed. During super-cluster localization step, for clusters having less than 3 non collinear, connected localized nodes are able to compute their position by considering nodes in adjacent clusters. In order to improve the overall position information, the Shadow Edges localization procedure is applied to the gateway nodes.

2.2.5 Recursive and Ad hoc Routing Based Localization in Wireless Sensor Networks

Kirci et al. [32] proposed a system that, when a mobile node needs to calculate its position, it tries to use reference nodes that are one hop away. If it does not have enough reference nodes that are one hop away to calculate its position, then it looks for multi-hop away reference nodes that will serve as distance reference nodes. To do so, the sensor node uses its neighbours that can relay using the OLSR [17] by routing the information of other ad hoc nodes that are multi-hop away from the sensor node. When a distance reference node is found, the estimated distance to the sensor node is calculated using the methods DV-hop and DV-distance. The ad hoc routing is necessary to search the multi-hop node that knows its position, and the OLSR routing is used to offer the position calculation of the nodes. Finally, the recursion based localization process will be completed by offering to estimate distances of nodes that are multi-hop away from the first node that already calculated its position.

2.2.6 A Recursive Shortest Path Routing Algorithm With Application for Wireless Sensor Network Localization

Cota-Ruiz et al. [19] presented a routing algorithm that can be used in the field of centralized range-based localization schemes. Given two non-neighbouring sensors, an unknown sensor and an anchor node, and only using the network connectivity, all evaluated shortest paths with the minimum number of hops are averaged to obtain a final distance estimate. This process can be repeated with different anchor nodes in order to estimate the unknown node position using both distance estimates and the absolute positions of anchors. To estimate distances between two non-neighbouring sensors, the proposed algorithms follows two steps: to find all paths with minimum

hops between the non-neighbouring sensors and to obtain the length of the paths based on the known one-hop distance. The non-neighbouring distance estimate is calculated as the mean of all evaluated path distances.

2.3 Duty-Cycling

Duty-Cycling in WSN [6, 28, 41] can be achieved in two different and complementary approaches: Topology Control and Power Management. Topology Control approaches [14, 15, 25] exploits node redundancy, selecting a minimum subset of sensor nodes to remain active for maintaining connectivity. On the other hand, Power Management approaches [43, 49] refers to active nodes switching off the radio when there is no network activity, alternating between sleep and awake periods.

2.3.1 Connected k-Neighbourhood (CKN)

CKN is an algorithm proposed by Nath and Gibbons [37] to keep at least $min(k, d)$ nodes awake in a sleep scheduling approach, where k is the number of connected nodes and d is the degree of each node. A sleep scheduler selects a subset of nodes to remain awake in a given epoch, the remaining nodes are set to a sleep state. The subset of awake nodes changes from epoch to epoch in order to improve the network life-time. The proposed algorithm generates a connected graph every epoch, enabling nodes to send and receive packets from each other within the network.

2.3.2 An Energy-efficient Coordination Algorithm (Span)

Span is a topology control protocol proposed by Chen et al. [15], nodes running Span make local decisions on whether to sleep, or to stay awake as a coordinator, performing multi-hop routing. To determine if a non-coordinator node should become

a coordinator or not, the node uses the following coordinator eligibility rule: using information gathered from local broadcast messages, a non-coordinator node checks if two of its neighbours cannot reach each other either directly or via one or two coordinators, if that is the case, the node should become a coordinator in order to maintain the network connectivity. The coordinator withdraw phase follows the same idea, a node checks if it should withdraw as a coordinator if every pair of its neighbours can reach each other either directly or via one or two other coordinators. According to the authors, Span preserves network connectivity and capacity, it also decreases latency, while providing significant energy savings.

2.3.3 Adaptive Self-Configuring sEnsor Networks Topologies (ASCENT)

Cerpa et al. [14] proposed Adaptive Self-Configuring sEnsor Networks Topologies (ASCENT). A node running ASCENT decides whether to become active or continue to sleep based on information about connectivity and packet loss that are measured locally by the node itself. Initially, there are only a few active nodes in the network, the remaining nodes keep listening but not transmitting messages. When the sink gets very high packet loss from the source, it starts sending help messages to signal passive neighbours to join the network. Upon receiving a help message, the passive node decides if it is going to join the network. As soon as it decides to join, the node signals the existence of a new active neighbour to the other passive nodes and it starts transmitting and receiving packets. This process repeats until the packet loss is reduced. ASCENT limits the packets loss due to collisions because the node density is taken into account as a parameter and has good scalability properties.

2.3.4 Sparse Topology and Energy Management (STEM)

Sparse Topology and Energy Management (STEM) is a technique proposed by Schurgers et al. [43]. It was notice that most of the time, the network is only monitoring the environment in order to an event to happen. Therefore, STEM uses two different radio signals. Each node periodically turns on its wakeup radio for a short time to listen if there is any neighbour trying to communicate with it. When a source node needs to communicate with a neighbouring node, it sends a stream of beacons on the wakeup channel. Upon receiving a beacon, the target node sends a wakeup acknowledgement and turns on its data radio. The actual data packets are transmitted through this radio. After the data transmissions have ended, the node turns its data radio off again. Results shows that STEM can reduce the energy consumption of the network, however, it also increases the setup latency.

2.3.5 Pipelined Tone Wakeup (PTW)

A Pipelined Tone Wakeup (PTW) scheme for WSN is proposed by Yang et al. [49]. Just like STEM, PTW works with two different channels for transmitting wakeup beacons and the actual data packets. The difference relies on what happens after receiving wakeup beacons, the wakeup procedure is pipelined with the packet transmission. When a node receives a beacon, it sends a wakeup acknowledgment and turns on its data radio, at the same time, the receiver node will send a wakeup beacon to wake up all its neighbours. The pipeline process reduces the wakeup latency and, consequently, the overall message latency.

Chapter 3

The Proposed ECKN: Joint Position Estimation, Packet Routing and Sleep Scheduling

3.1 Sensor Node Position Estimation

In a WSN where a mobile sink collects information from sensor nodes, the sink is frequently not in the range of a node that has a packet to be forwarded. Therefore, the forwarding node has to find a path through other sensor nodes in order to reach the sink. Location-awareness facilitates in the decision of finding the best neighbour to forward the packet. Being aware of its neighbours' positions and the sink position, the forwarding node can send the packet to a neighbour which is closer to the sink. As mentioned earlier, the usage of a GPS module is unfeasible in small, cheap, low-power devices. Thus, ordinary wireless sensor nodes cannot afford to carry GPS modules. As the sink moves, it shares beacons of its new position so that sensor nodes have the most up-to-date position to perform the position-based packet routing. The sensor

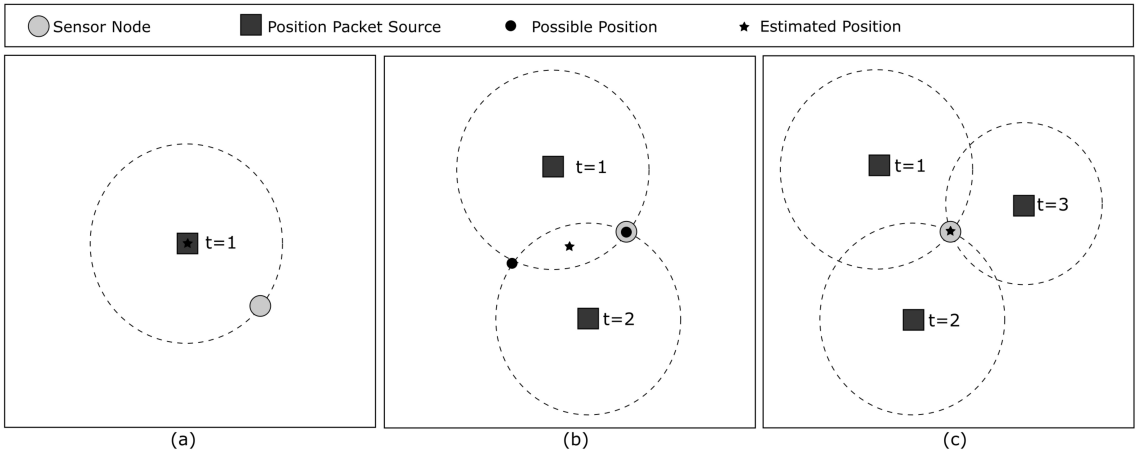

Figure 3.1: Overview of how the position of the sensor node is estimated. (a) With one position packet source. (b) With two position packet sources. (c) With three position packet sources.

nodes can take advantage of these position packets in order to estimate their own position.

In the proposed position estimation algorithm, a sensor node can estimate its own position not only by using packets from the sink, but also from position information shared by neighbouring nodes. To this end, once a node estimates its position, it shares its own position with its neighbour nodes, helping them to estimate more accurate positions. It is important to notice that the packets with position information from the sink are more reliable since the position of a node is estimated based on the sink position, which in turn is based on its GPS position. Therefore, a sensor node will always give preference to the position information received from a sink, which will, eventually, replace the position received from a neighbour node. At least three position samples are needed to estimate the position of a node more accurately. However, it is still possible to estimate the position of a node by using only one or two position samples as can be seen in Figure 3.1.

The estimation of a node's position can be performed according to the number of position samples received from position sources:

- one position sample: the sensor node will assume that its position is the same as the one received in the packet. It is not accurate but with this information it is possible to at least determine the region where an event occurred.

- two position samples: when calculating the intersection between two position packets, I have two possible positions for the node. Since I don't have any other information, the approximate position of the sensor node is the midpoint between these two calculated positions.

- three position samples: now a node can estimate a more accurate position and it becomes a qualified node. Using the trilateration algorithm, it is possible to estimate the sensor node position by finding the intersection between these three position samples.

When the network starts operating, the positions of sensor nodes will be inaccurate due to the lack of position packets to estimate their positions. However, node localization will eventually become more and more accurate over time because more position packets will be disseminated in the sensor field as the sink moves. Afterwards, it will be possible to start tracking events that occur in the area and forward them to the sink.

3.1.1 Uncertainties in GPS Positioning

Although trilateration is a viable solution for the localization problem, GPS uncertainty makes position estimation more complex. When receiving position packets and calculating the intersection between them, there is no common intersection point among the packets due to the GPS error. Instead, each intersection will generate two intersection points and the sensor node will have to decide which one to use as a reference point to estimate its position. Figure 3.2 shows an example of how a

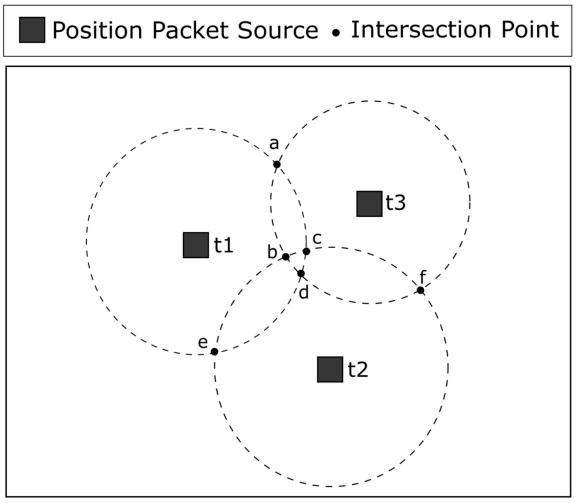

Figure 3.2: Intersection Points among the Position Packets.

sensor node calculates the intersection points. It receives three position packets from the sink at times $t1$, $t2$ and $t3$. The intersection between $t1$ and $t2$ generates the intersection points c and e. The intersection between $t1$ and $t3$ generates the intersection points a and d. Similarly, the intersection between $t2$ and $t3$ generates the intersection points b and f. I propose two algorithms to find out which intersection points to use as reference points: Distance and Closeness algorithms. Afterwards, the sensor node can simply estimate its position calculating the average of the reference points (a, b, c) obtained according to Equation 3.1.

$$P(x, y) = \big((x_a + x_b + x_c)/3, (y_a + y_b + y_c)/3 \big) \tag{3.1}$$

Distance

In order to obtain the reference points through the distance algorithm, the sensor node calculates the distance between each pair of intersection points. For instance, consider the set of pairs (c, e), (a, d) and (b, f) of Figure 3.2. When deciding between the points c and e, the sensor node calculates their distances to all other intersection

points, that is a, b, d and f. The point with smallest distance (point c in the example) is set as a reference point as can be seen in Equation 3.2.

$$\forall p \in s : min\Big(\sum \big(d(i,a) + d(i,b)\big)\Big) \;\; \forall o(a,b) \in s \;\; and \;\; o \neq p \tag{3.2}$$

where p is the pair of intersection points; s is the set of pairs; $d(x,y)$ is the distance between two intersection points x and y; i is each intersection point of the pair; and o is other pairs of intersection points a and b.

Closeness

My second algorithm to obtain the reference points is called Closeness. In this algorithm, the sensor node creates every possible combination among the intersection point pairs. In Figure 3.2, the sets created after this combination are: (a,b,c), (a,b,e), (a,c,f), (a,e,f), (b,c,d), (b,d,e), (c,d,f) and (d,e,f). Subsequently, the sensor node calculates the distance between all pairs of points in each set. Then, it sums the distances within a set. The intersection points in the set with the smallest sum are considered as reference points as can be seen in Equation 3.3.

$$rp = ip(a,b,c) : ip \in s \;\; and \;\; ip \;\; has \;\; min(d(a,b) + d(a,c) + d(b,c)) \tag{3.3}$$

where rp is the set of references points; ip is each combination of three intersection points; a, b, and c, s is the set of these combinations; and $d(x,y)$ is the distance between two intersection points x and y.

3.1.2 The Position Packet Validation Algorithm

Since the mobile sink is more powerful, it can carry a GPS module. Thus, it can provide position packets to sensor nodes in order to aid them to estimate their own positions. Furthermore, in my scenario, the random mobility adopted by the sink implies that some position packets might not be good enough to be used in the trilateration algorithm, as can be seen in Figure 3.3. Figure 3.3(a) shows a sensor node receiving two position packets at times $t1$ and $t2$ from opposite directions. In this scenario, the intersection between these packets generates a single point. However, GPS error may occur and it may shift $t1$ slightly to the left and/or $t2$ slightly to the right. In this case, the circles would not intersect, making it impossible to estimate the sensor node position. Thus, upon receiving a new position packet, the sensor node will firstly validate if this packet position circle intersects with the positions previously validated. If so, the next phase of the validation will be performed. Otherwise, the packet is discarded.

In Figure 3.3(b), it is possible to see that if I divide the network area into four quadrants with the sensor node positioned at the center, all position packets shown belong to the same quadrant. GPS error produces a poor position estimation in this case, essentially because when one circumference is contained within another, a small deviation caused by the error results in a large discrepancy of intersection points. Therefore, I propose three algorithms to validate the position packet and avoid the scenario described above: Position Distance, Circle Limit and a Hybrid solution. After the position packet is validated and accepted, i.e., the position is a good candidate to be used in the trilateration algorithm, the sensor node stores its information for future estimations.

Even after having received three good position packet candidates, the sensor node keeps receiving this type of packets in order to try to improve its position estimation.

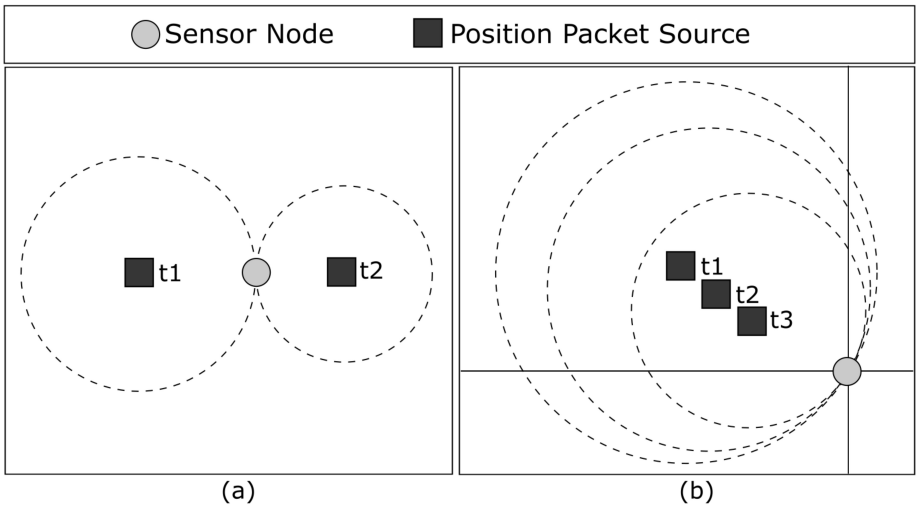

Figure 3.3: (a) Position packets in opposite sides may not intersect due to GPS error. (b) Poor trilateration due to all position packets laying on the same quadrant.

Essentially, it uses the RSSI of the newly received packet and calculates the distance between its estimated position and the position of the packet source. If they do not match (within a certain threshold), the sensor node validates the packet and checks if replacing one of its current position packets with the new one will result in a more accurate estimation. If so, the replacement is done permanently.

Position Distance

When executing the Position Distance algorithm, a sensor node stores the information of the first received position packet. When the next position packets are received, the sensor node calculates the distance between the source position of the newly received packet and the source position of each packet that was already stored. If all of these distances exceed a predefined threshold, that is, Equation 3.4 is true, the packet is accepted and the sensor node stores its information to be used in future estimations. If at least one of the calculated distances does not exceed the threshold, the received position packet is considered invalid and it is dropped. Figure 3.4(a) shows a sensor

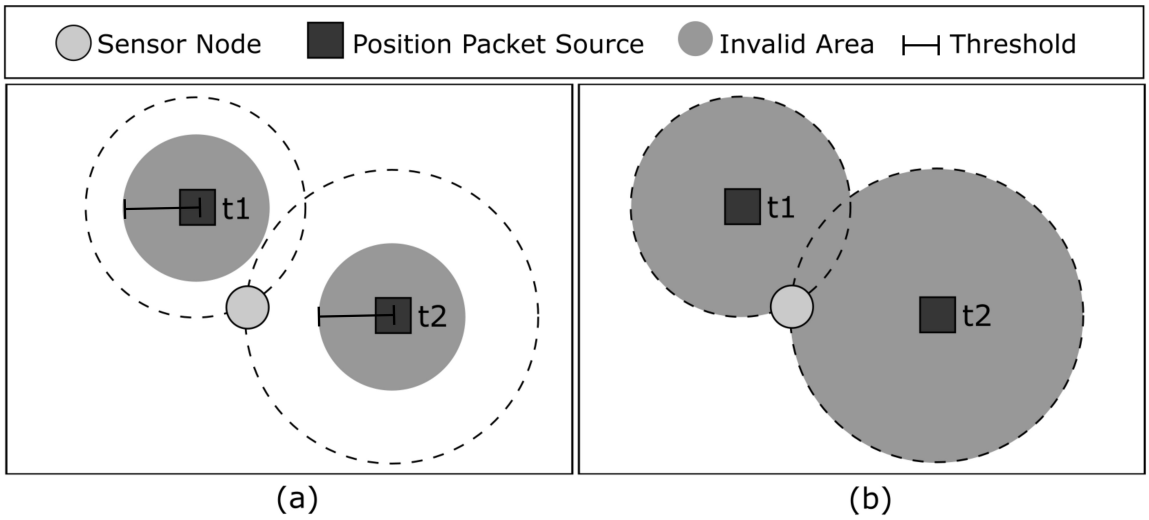

Figure 3.4: (a) Invalid area for position packets according to Position Distance algorithm. (b) Invalid area for position packets according to Circle Limit algorithm.

node that received, validated and stored two position packets at times $t1$ and $t2$. The shaded region represents the area where other position packets will be considered invalid based on the threshold, so that I avoid the issue depicted in Figure 3.3(b). Any packet received from a position outside this area will be accepted.

$$\forall p \in rp : d(p, r) > t \tag{3.4}$$

where rp represents the set of received packets; r is the newly received packet; $d(p, r)$ is the distance between p and r; and t is the predefined threshold.

Circle Limit

When executing the Circle Limit algorithm, a sensor node also stores the information of the first received position packet. Upon receiving subsequent position packets, the sensor node checks whether the source position of the received packet is within the circle of each stored packet. The circle radius is defined by the RSSI of the packet. The sensor node also checks if each source position of the stored position packets are

within the circle of the newly received packet. If none of these cases occur, that is Equation 3.5 is true, the packet is accepted and the node stores its information to be used in future estimations. Otherwise, the received position packet is considered invalid and it is dropped. Figure 3.4(b) shows a sensor node that received, validated and stored two position packets at times $t1$ and $t2$. The shaded regions represent the circles calculated based on the RSSI. If the received position packet is within the shaded area, or its circle contains one of the validated source position packets, it is considered invalid. Otherwise, the received packet is accepted.

$$\forall p \in rp : r \notin c_p \text{ and } p \notin c_r \tag{3.5}$$

where rp represents the set of received packets; r is the newly received packet; c_p and c_r are the circles of p and r, respectively.

Hybrid

Position Distance does not guarantee that a quadrant will contain a single position packet, but it improves the probability of that occurring. On the other hand, Circle Limit does guarantee that a quadrant will have only one position packet. However, it reduces the chances of validating and accepting new position packets. This implies that a node may take longer to estimate an accurate position. Therefore, I propose a hybrid algorithm that takes advantage of both algorithms. Initially, a sensor node will run the Position Distance algorithm until it is considered qualified, i.e., the node has received and validated three position packets. Afterwards, the node switches to Circle Limit in order to improve its position estimation. Thus, a node can quickly estimate an acceptable position initially and then improve it over time.

3.2 Event Packet Forwarding

The proposed packet forwarding algorithm aims to find the best node to forward the packet to based on the GGF algorithm using an utility function that considers characteristics of the network such as power level and distance to the sink. Power level represents how much energy the sensor node still has. More energy remaining means that the node is the best neighbour to forward the packet to because the chance of shutting down the node before forwarding the packet to the next hop is lower. Distance to the sink represents the ratio between the distance of the receiver node to the sink and the distance of the sender node to the sink. The lower this ratio is, the better it is to send the packet to the node because it will be closer to the sink. This ratio is more important when the packet is close to the sink because when it is far, there are more than one good route to forward the packet. This situation can be seen in Figure 3.5.

After running the localization estimation algorithm, each sensor node stores the position of its neighbours and the sink. As the sink moves, it shares its new positions, and the sensor nodes replace the stored sink position to always have the most updated position. The sink position packet will also have a Time to Live (TTL), meaning that when a sensor node receives this packet, it will decrease TTL by one and, if it is still greater than 0, it will forward the packet to its neighbours, helping to update the sink position for nodes that are further away from the sink. A description of the sink position packet fields can be found in Table 3.1. Similarly, a sensor node shares its current battery level, its address, its position, and the accuracy of its position, i.e., the position was estimated using three accurate position packets. A description of the contents of a sensor node position packet can be found in Table 3.2.

Using this information, it is possible to define a utility function expressed by

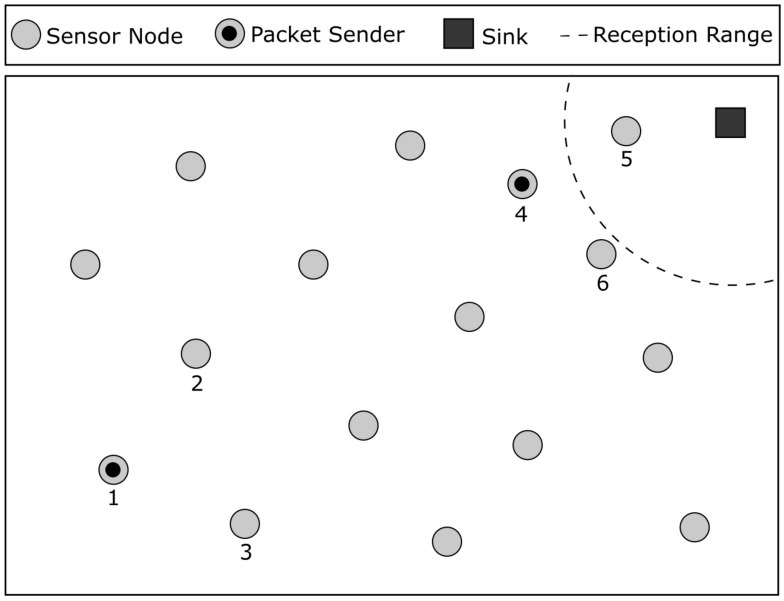

Figure 3.5: Distance to sink. Sensor Node 1 can choose between 2 and 3 and the distance is not a great factor to make the decision. On another hand, it is more important that Node 4 forwards the packet to Sensor Node 5 than to Sensor Node 6 because node 5 is in the sink reception range.

Table 3.1: Sink Position Packet

Field	**Type**	**Description**
sqnNum	integer	Sequence Number of the packet, the nodes are going to update the position of the sink only if the sqnNum is bigger than the previous received.
posX	double	Position of the sink in X-axis
posY	double	Position of the sink in Y-axis
ttl	integer	The nodes are going to forward the sink position packet to its neighbourhood until ttl is 0.

Table 3.2: Sensor Node Position Packet

Field	**Type**	**Description**
address	string	The address of the sensor node.
battery	double	The power remaining in the battery of the sensor node.
positionX	double	Position of the sensor node in the X-axis.
positionY	double	Position of the sensor node in the Y-axis.
accurate	Boolean	Whether or not the position of the sensor node is accurate.

Equation 3.6 to estimate the best node to forward the packet to.

$$U_i = w_p P_i + w_d(1 - D_i/D) \tag{3.6}$$

$$w_p + w_d = 1 \tag{3.7}$$

where i is the ith neighbour node; w_p is the weight for power attribute; P_i is the power of neighbour node i; w_d is the weight for distance attribute; D_i is the distance to the sink of neighbour node i; D is the distance to the sink of the sender node.

By Equation 3.7, I have that the sum of the weights must be 1. Since the routing algorithm is based on GGF, the most important attribute in Equation 3.6 is the distance to the sink. Thus, the proposed values for the weights are $w_p = 0.25$ and $w_d = 0.75$, in order to prioritize the position of the sensor nodes.

3.2.1 Sink Position Bridges

The proposed packet forwarding algorithm is performed based on the position of the sink. Therefore, the sensor nodes need to know the sink position and they need to update this information as the sink moves.

As mentioned in Section 3.2, the sink broadcasts its position with a TTL. After receiving this packet, the sensor node decreases the TTL by one and checks if it is greater than zero. If so, it forwards this packet to neighbouring nodes.

Since the sink has random mobility, it is possible that an event packet travels a much longer path to reach the sink depending on how the sink moves. Figure 3.6(b) shows one example of this path. Sensor Node 1 will forward the packet following the sink trail, thus the path that the event packet will take to reach the sink is $1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 5 \rightarrow 6 \rightarrow 7 \rightarrow 8$. However, there is a possible shortest path for

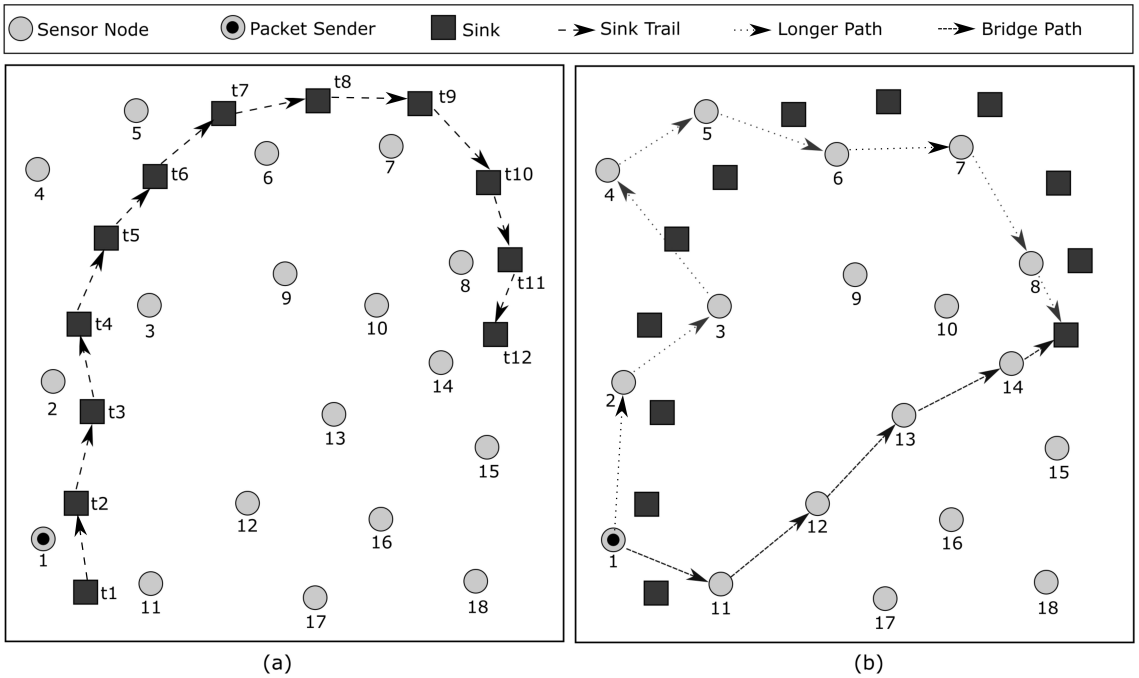

Figure 3.6: (a) Bridges depend on the sink movement. (b) A bridge is created when sink calculates that there is a shortest path between its current position and a previous position.

this event packet, which is $1 \rightarrow 11 \rightarrow 12 \rightarrow 13 \rightarrow 14$.

In order for the sink to find the shortest path, i.e., bridge, between its previous positions and its current position, the sink keeps track of a certain number of previous positions to estimate the existence of a bridge. As the sink moves, previous positions are stored so that its traveled distance can be estimated. For instance, Figure 3.6(a) shows where the sink saves its position, which is done every x meters. When the sink is at $t12$, it can easily calculate how much it has traveled. For instance, when the sink is at $t11$, it has traveled x, at $t10$ it has traveled $2x$, at $t9$ it has traveled $3x$, and so on.

Once the distance travelled by the sink is known, I can compare the Euclidean distance between the current and previous positions. If the ratio of those two positions is less than a predefined threshold, then there is a shortest path and a bridge is created.

In order to create a bridge, the sink sends its packet position with a bigger TTL. This means that this position will reach the sensor nodes close to its previous position, helping them to find the shortest path by updating the sink position.

3.3 Sleep Scheduler

Network lifetime is a key point in WSNs. Due to the limited power in the sensor nodes, different approaches have been studied in order to prolong the network lifetime. Sleep scheduling is one of these approaches. In sleep scheduling method, sensor nodes work in duty-cycles, different nodes go to sleep in each epoch and the rest remain awake. When a sensor node is sleeping, it saves the most energy possible by reducing its transmission and reception channels and processing power. Although sleep scheduling is a viable solution to extend network lifetime, it may interfere in the packet routing. Depending on what nodes go to sleep, the network may get disconnected. Figure 3.7 shows an example of this scenario, nodes e and f are the only connection between nodes b and h, if both go to sleep in the same epoch, there would be no route between b and h.

In order to avoid scenarios as the one described above, Nath et al. [37] proposed an efficient decentralized sleep scheduling algorithm to reduce the number of awake nodes while maintaining the network connected. Their algorithm addresses the CKN problem, a NP-complete problem in graph theory: Given a constant k and an undirected graph $G = (V, E)$, find a subset of nodes $C \subset V$ such that C is a minimum connected k-neighbourhood. In CKN, (i) each node $v \in V$ has at least $m = min(k, d_v)$ neighbours from C, where d_v is the degree of v in G, and (ii) the nodes in C are connected. C is a minimum CKN if no CKN has a smaller number of nodes.

The near-optimal solution to the CKN problem is depicted in Algorithm 3.1, a

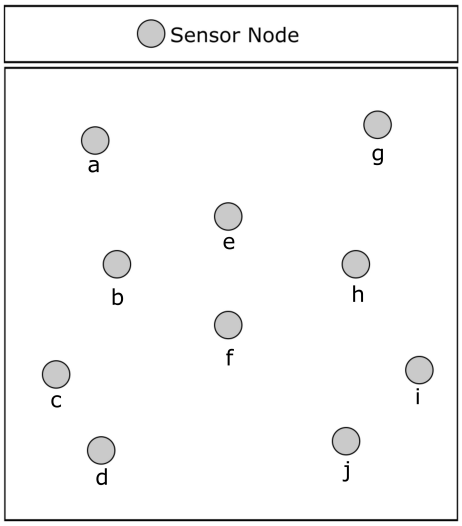

Figure 3.7: Sleep scheduling may disconnect the network if nodes e and f go to sleep in the same epoch.

comprehensive explanation of the algorithm is detailed in their work [37]. In order to determine if it is going to sleep, each node picks a random rank from a random number generator. This rank is then compared with the neighbourhood. Nodes with larger ranks have higher probability of going to sleep. In order to balance the network lifetime among the nodes, instead of using a random rank to decide which nodes are going to sleep, Yuan et al. [51] proposed to use the residual energy information of the nodes as the parameter to decide whether a node to be awake or sleeping. In my solution, I take advantage that the energy information is shared in order to perform the packet routing and use it to get the rank for each node using Equation 3.8. Thereafter, having the rank information, sensor nodes proceed to estimate whether they are staying awake or going to sleep using the same algorithm as the one proposed by Nath et al. [37]. The proposed ECKN algorithm is depicted in Algorithm 3.2

$$rank_u = 1 - energy_u \tag{3.8}$$

where $rank_u$ is the rank that node u will use to determine if it is going to remain

awake or go to sleep; and $energy_u$ is the residual energy of node u.

Algorithm 3.1 CONNECTED K-NEIGHBOURHOOD (CKN) (* Run the following at each node u *)

1: Pick a random rank $rank_u$.
2: Broadcast $rank_u$ and receive the ranks of its currently awake neighbours N_u. Let R_u be the set of these ranks.
3: Broadcast R_u and receive R_v, from each $v \in N_u$.
4: If $|N_u| < k$ or $|N_v| < k$ for any $v \in N_u$, remain awake. Return.
5: Compute $C_u = \{v | v \in N_u \text{ and } rank_v < rank_u\}$
6: Go to sleep if both the following conditions hold. Remain awake otherwise.

- Any two nodes in C_u are connected either directly themselves or indirectly through nodes within u's 2-hop neighbourhood that have $rank$ less than $rank_u$.
- Any node in N_u has at least k neighbours from C_u.

7: Return.

Algorithm 3.2 ENERGY-AWARE CONNECTED K-NEIGHBOURHOOD (ECKN) (* Run the following at each node u *)

1: Let $rank_u = 1 - energy_u$.
2: Broadcast $rank_u$ and receive the ranks of its currently awake neighbours N_u. Let R_u be the set of these ranks.
3: Broadcast R_u and receive R_v, from each $v \in N_u$.
4: If $|N_u| < k$ or $|N_v| < k$ for any $v \in N_u$, remain awake. Return.
5: Compute $C_u = \{v | v \in N_u \text{ and } rank_v < rank_u\}$
6: Go to sleep if both the following conditions hold. Remain awake otherwise.

- Any two nodes in C_u are connected either directly themselves or indirectly through nodes within u's 2-hop neighbourhood that have $rank$ less than $rank_u$.
- Any node in N_u has at least k neighbours from C_u.

7: Return.

It was observed that sensor nodes close to the sink tend to consume more energy since they receive packets from the whole network to forward them to the sink. On the other hand, nodes far from the sink tend to consume less energy, because the traffic of packets is much lower. Therefore, thinking about this situation, I propose different

values for k depending on how far a sensor node is from the sink. I divide the network in 3 regions: high, medium, and low traffic. Nodes in the high traffic region will have a larger value for k, because they receive packets from high, medium, and low regions. Having a larger value for k implies that more nodes are going to stay awake in that region and the workload can be distributed. Nodes in the medium traffic region will have a medium value for k, because they receive packets from medium and low regions only. Nodes in the low traffic region will have smaller values for k, since they are far from the sink, they are just going to receive packets from nodes in the low traffic region as well. In this region, more nodes can go to sleep in order to save energy.

In the previous section, I showed how the sink and sensor nodes estimate and share their position. Having this information, sensor nodes can calculate in which region they are by using the distance between them and the sink. Figure 3.8 shows an example of how the network is divided. Nodes in the high traffic region, which are closer to the sink, have the value $k1$ for k, nodes in the medium traffic region have the value $k2$ for k, and nodes in the low traffic region have the value $k3$ for k. I have that $k1$ is greater than $k2$, and $k2$ is greater than $k3$. It is possible to identify in the figure that the network is denser in the region that contains the sink due to the high traffic of packets that this region is going to receive. In the same way, the region far from the sink is more sparse since the traffic of packets is lower far from the sink.

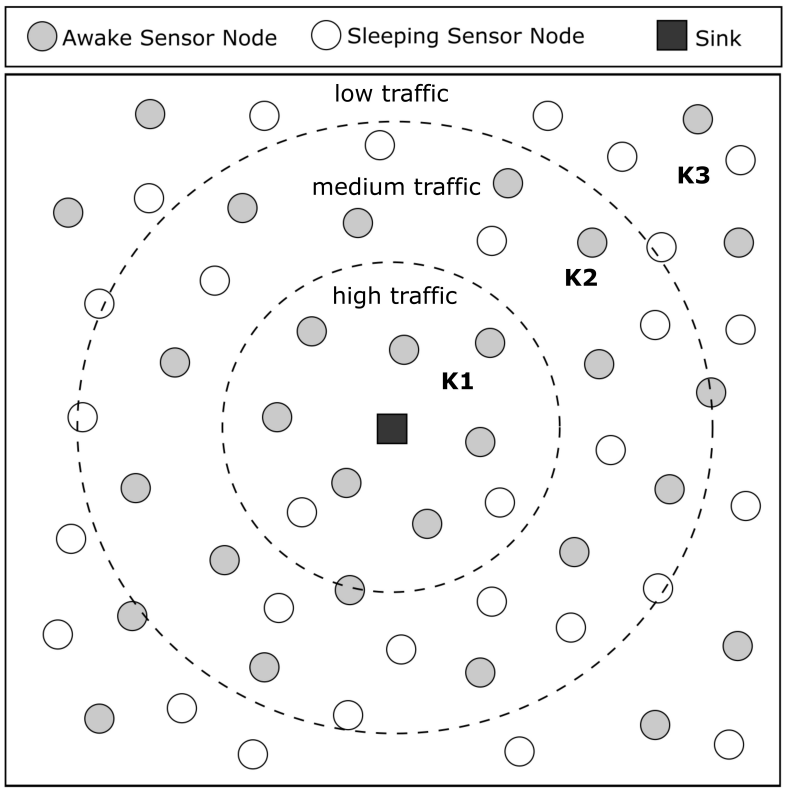

Figure 3.8: The closer the nodes are from the sink, the higher is the value of K. This way, the network will be more dense close to the sink, where there are more packet transmissions.

Chapter 4

Performance Evaluation

An extensive set of simulations was conducted to evaluate the performance of the proposed schemes. The performance metrics are position estimation accuracy, packet delivery ratio, and network overhead.

Simulations were implemented in OMNeT++ [1] and parameters can be found in Table 4.1. The number of sensor nodes is 200. Network size varies from 500x500 m^2 to 1000x1000 m^2 in order to evaluate the schemes in sparse and dense networks. The mobile sink starts at the center of the area, i.e., (500, 500), and its average speed is 10m/s, e.g., a drone, following the random waypoint as its mobility model [42]. The sink sends its position to nodes every 5s. Transmission range of the sink and nodes is 185m using breakpoint path loss [20]. The threshold for Position Distance algorithm is 50m which presented the best performance in my scenario. I used IEEE 802.15.4 protocol as the MAC Layer. Simulations were run 10 times following the Student's t-distribution [22]. I compare my algorithms with trilateration without packet validation in order to show the performance improvement.

[1] OMNeT++ (https://omnetpp.org/) is an extensible, modular, component-based C++ simulation library and framework, primarily for building network simulators.

Table 4.1: Evaluation Parameters

Parameter	**Value**
Number of Sensor nodes	200
Network Size	500x500m^2 to 1500x1500 m^2
Number of Sinks	1
Sink Start Position	(500, 500)
Sink Avg Speed	10m/s
Sink Mobility Model	Random Waypoint
Sink Beacon Frequency	5s
Transmission Range	185m
Position Distance Threshold	50m
MAC Layer	IEEE 802.15.4

4.1 Sensor Node Position Estimation Accuracy

Firstly, I evaluate the Distance and Closeness algorithms when selecting the most appropriate reference points. I simulate the same scenario for both algorithms together with the Position Distance algorithm to estimate node positions. Figure 4.1 shows the number of nodes within different position estimation error ranges. As can be seen, Closeness has more nodes in the small error range while Distance exceeds Closeness in larger error ranges. Thus, Closeness outperforms the Distance algorithm because it considers all possible combinations of intersection points and chooses the closest points within a combination set to be used as reference points. Therefore, the Closeness algorithm is used in the next simulations in order to obtain Reference Points.

My next simulation evaluates how long it takes a sensor node to become qualified, as shown in Figure 4.2. Both Position Distance and Hybrid algorithms have shown similar performance to the one with no validation, and they perform better than Circle Limit. This can be explained in Figure 3.4, where Position Distance has a smaller invalid area. This means that the probability of accepting a position packet is higher. Consequently, nodes become qualified faster. In addition, Hybrid performance is

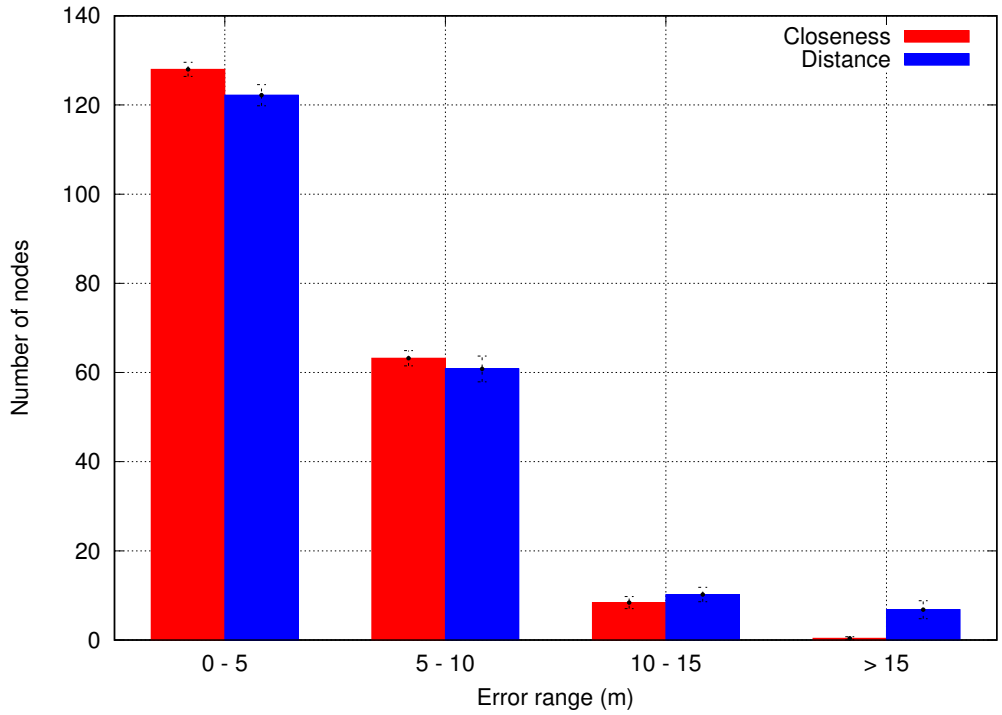

Figure 4.1: Impact of reference points algorithms in sensor node accuracy.

similar to the Position Distance algorithm because it uses the same algorithm until the node is considered qualified.

Figure 4.3 shows the average position estimation error in terms of the distance between the estimated position and the actual position for all nodes. Position Distance outperforms Circle Limit because it validates and accepts more position packets. Hybrid starts to perform better than Position Distance over time because after the sensor node is qualified, it starts running Circle Limit algorithm, which results in better estimations. Although the algorithm with no validation accepts every position packet, it is outperformed by the other algorithms because it does not avoid situations depicted in Figure 3.3, which results in inaccurate estimations.

Although Position Distance shows better performance in terms of position estimation error when considering all nodes, Circle Limit shows improved performance when I consider only qualified nodes, as shown in Figure 4.4. This happens because sce-

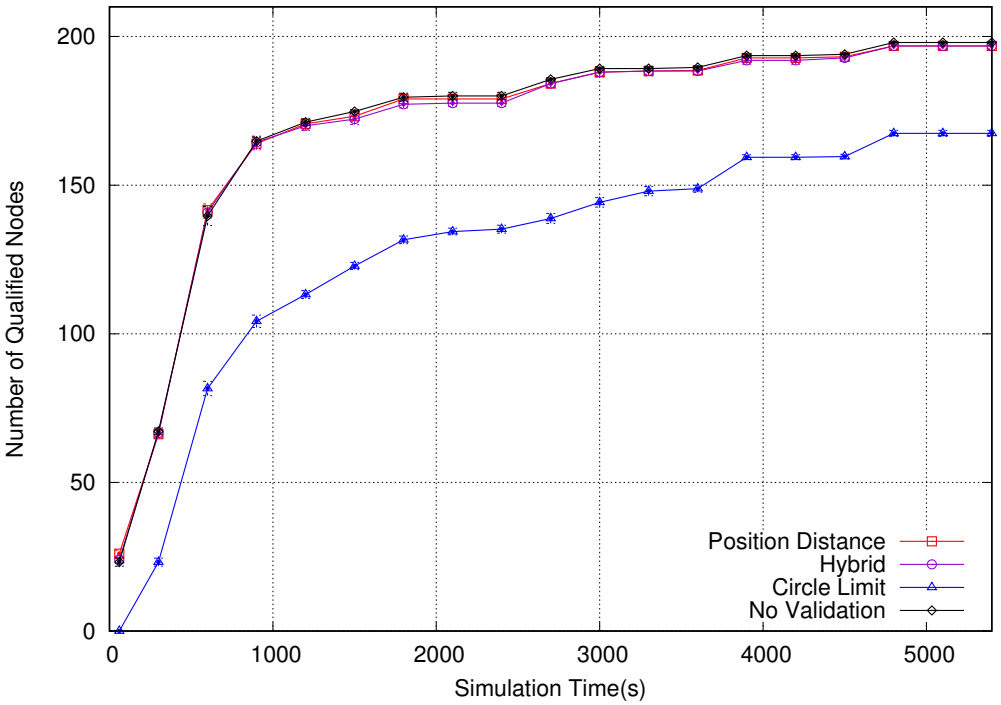

Figure 4.2: Number of qualified sensor nodes.

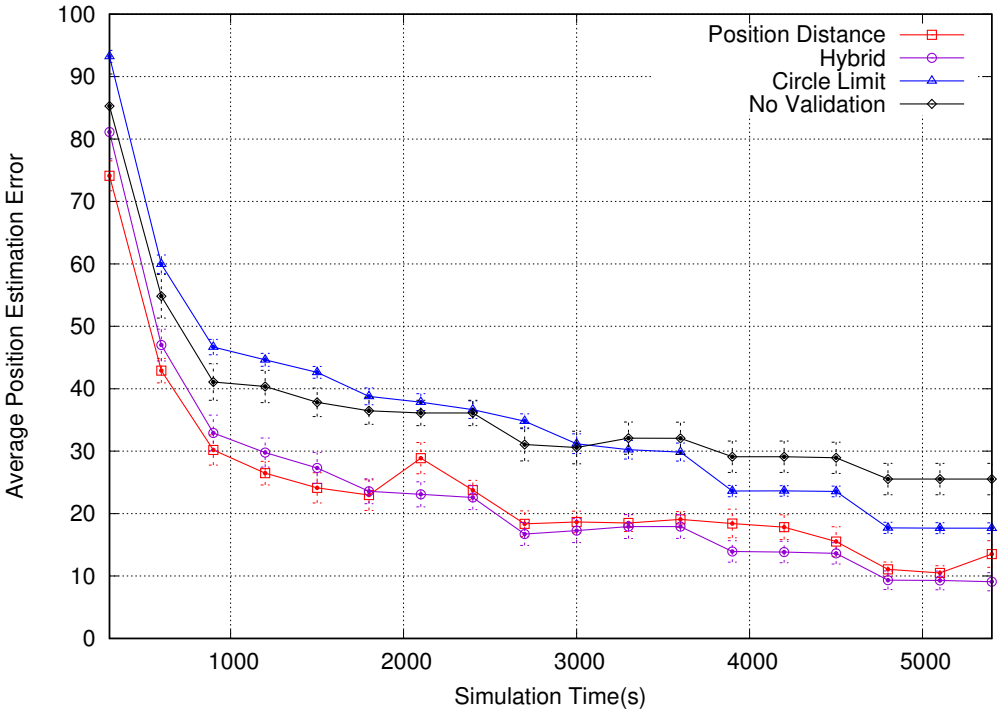

Figure 4.3: Average position estimation error for all sensor nodes.

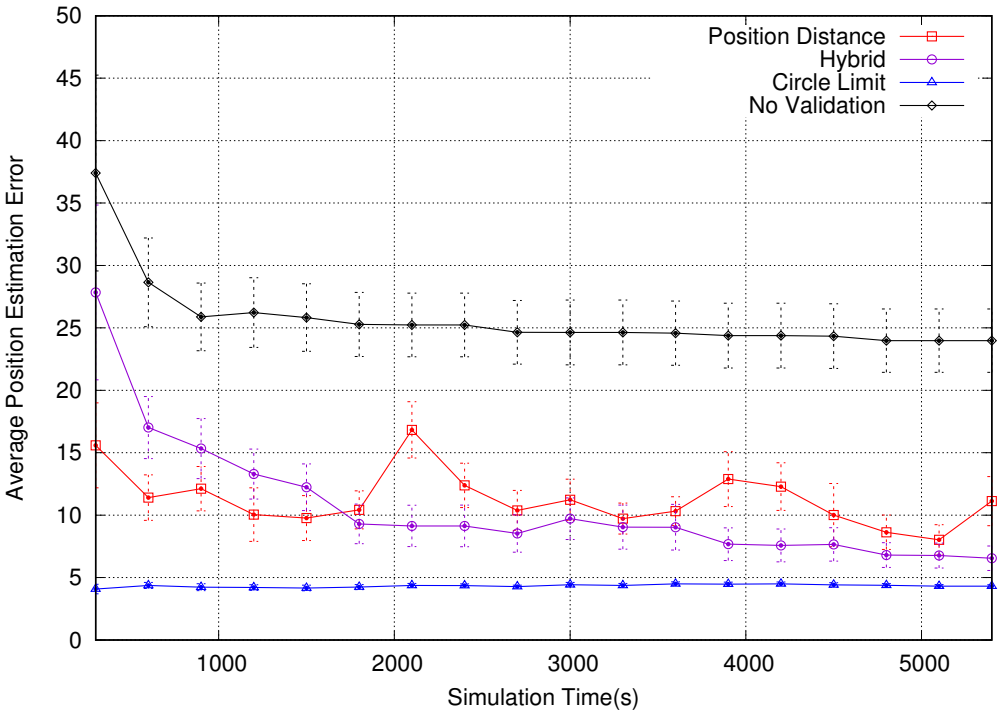

Figure 4.4: Average position estimation error for qualified nodes.

narios such as in Figure 3.3(b) are impossible to occur in Circle Limit, while Position Distance minimizes their occurrence. Hybrid outperforms Position Distance over time for the same reason that it starts running Circle Limit after having qualified nodes as discussed earlier. The algorithm with no validation is outperformed once again due to the same reason that it does not avoid situations depicted in Figure 3.3.

Position estimation outlier errors for qualified nodes is depicted in Figure 4.5. Circle Limit clearly shows better performance because it guarantees that a quadrant will only have a single position packet, which results in better estimation for qualified nodes. The Hybrid algorithm shows better results over time because it starts running Circle Limit algorithm. The algorithm with no validation and Position Distance show similar performance because both do not completely avoid scenarios such as in Figure 3.3(b).

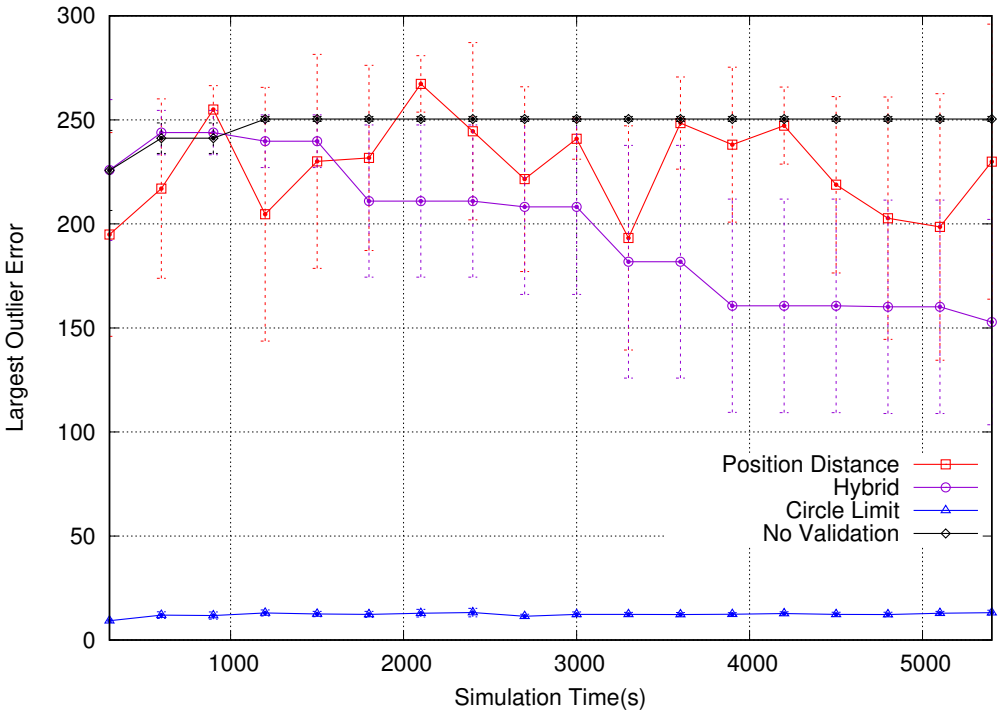

Figure 4.5: Largest outlier error.

4.2 Packet Routing

I evaluate the packet delivery success ratio by simulating the protocols in different network densities, five different sizes for the network were used: 500x500m^2, 750x750m^2, 1000x1000m^2, 1250x1250m^2 and 1500x1500m^2. As depicted in Figure 4.6, ECKN with bridges performs better than ECKN without bridges and GGF, the reason is that ECKN creates bridges that shortens the distance between the source node and the sink, increasing the probability of a packet to be delivered. GGF performs better than ECKN without bridges, the main reason is that ECKN takes into consideration the energy remaining in the node besides its position, therefore, the packet may take a longer path, increasing the probability of dropping this packet. Another important point to analyze is that 1000x1000m^2 network size presents the best average delivery ratio, this is because smaller areas present lots of collision, therefore event packets

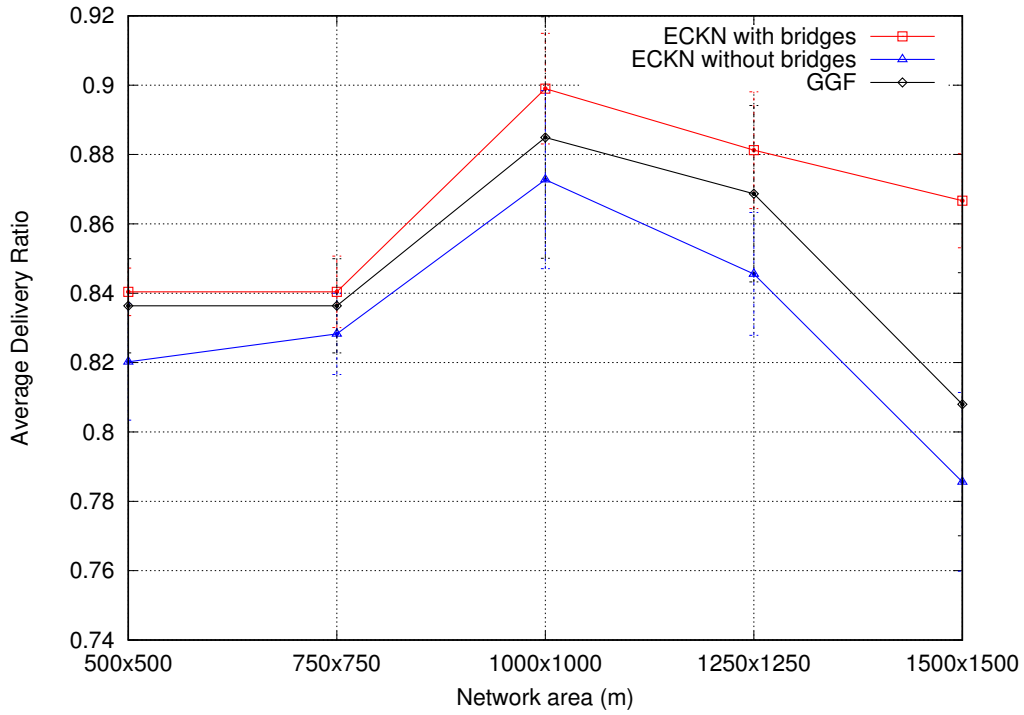

Figure 4.6: Delivery Ratio of GGF, ECKN with and without Bridges versus number of sensor nodes in the area.

may get lost during the routing. On the other hand, larger areas present blind spots, making the routing not feasible, causing the event packet to be dropped.

The packet delivery success is directly related to the number of hops that an event packet takes to get from the source node to the sink. Figure 4.7 shows the comparison between ECKN and GGF. ECKN with bridges presents the smaller average number of hops, this is explained by the bridges that are created. The bridges shorten the path between the source node and the sink, reducing the number of hops necessary for the event packet to be delivered. GGF and ECKN without bridges present similar performance, with a slight advantage for the first one. The justification for this advantage is that, in GGF, the sensor node always chooses the closest neighbour to the sink to forward the packet while the energy remaining is also considered when performing ECKN. Due to this reason, ECKN without bridges sometimes takes a

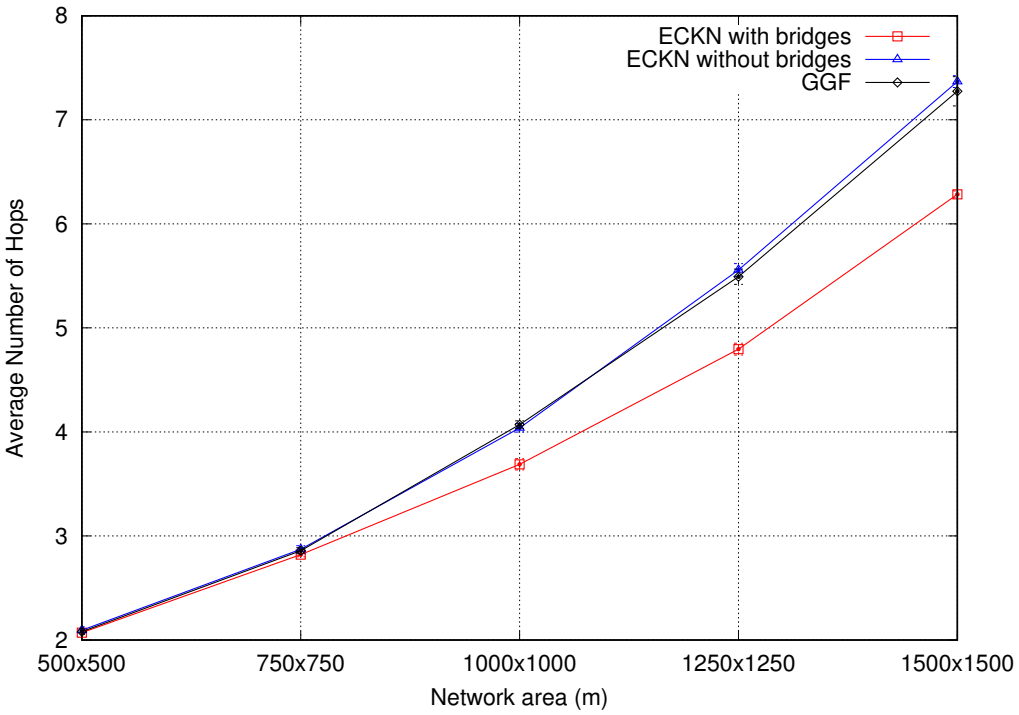

Figure 4.7: Number of hops of GGF, LOGR with and without Bridges to get to the sink versus number of sensor nodes in the area.

longer path to reach the sink.

The network overhead is also evaluated in my simulations. As can be seen in Figure 4.8, GGF and ECKN without bridges present same behaviour while ECK with bridges presents a larger overhead. This is explained by the larger TTL that ECKN with bridges uses when creating a bridge. While GGF and ECKN without bridges have the same TTL for every position packet from the sink, ECKN with bridges calculates if it is possible to shorten the path between an area and the position of the sink, if positive, it increases the TTL of the position packet in order to reach a region farther away. This causes the position packet to be resent more times, increasing the network overhead.

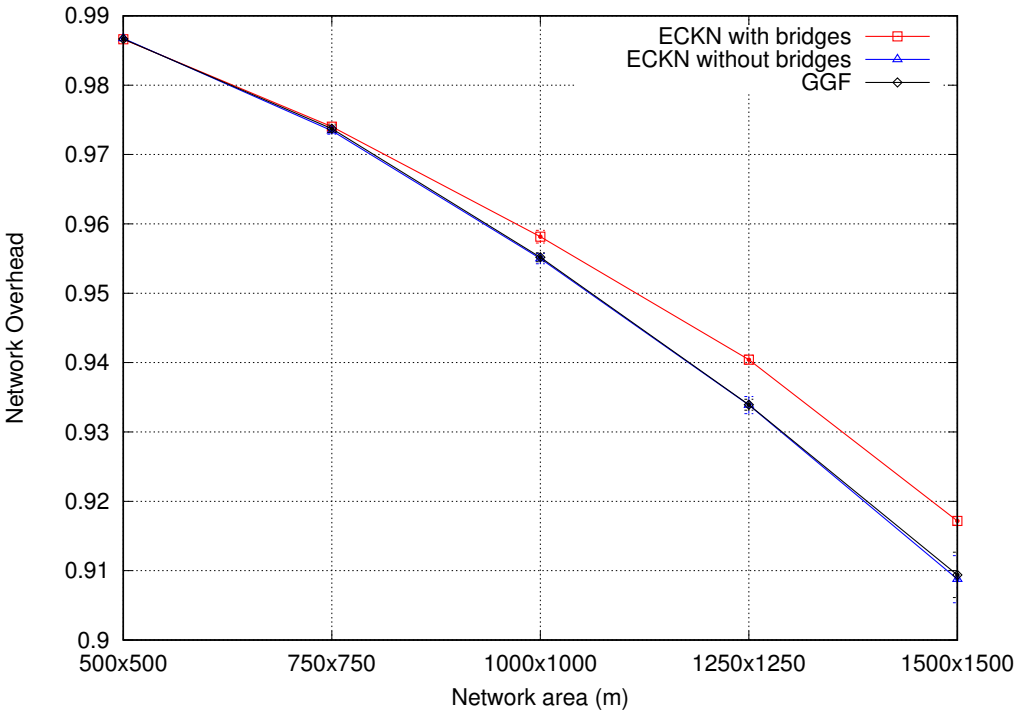

Figure 4.8: Network Overhead.

4.3 Network Lifetime

In this section I evaluate the impact of my proposed sleep scheduler in the network lifetime. Firstly, I evaluate the impact of using the residual energy of the node instead of a random number as rank in the CKN algorithm. I also evaluate my ECKN algorithm. The proposed solution has $k = 3$ for nodes close to the sink, $k = 2$ for nodes that are in intermediate region, and $k = 1$ for nodes farther from the sink. I compare my algorithm with the solution having no sleep scheduler and with CKN having $k = 3$, $k = 2$, and $k = 1$.

Figure 4.9 shows the network lifetime using random numbers and residual energy as rank. As can be seen, having random numbers as ranks shows better performance than using the energy remaining. This is explained by how the CKN algorithm works. In order to go to sleep, a sensor node has to follow two conditions depicted in Step 6

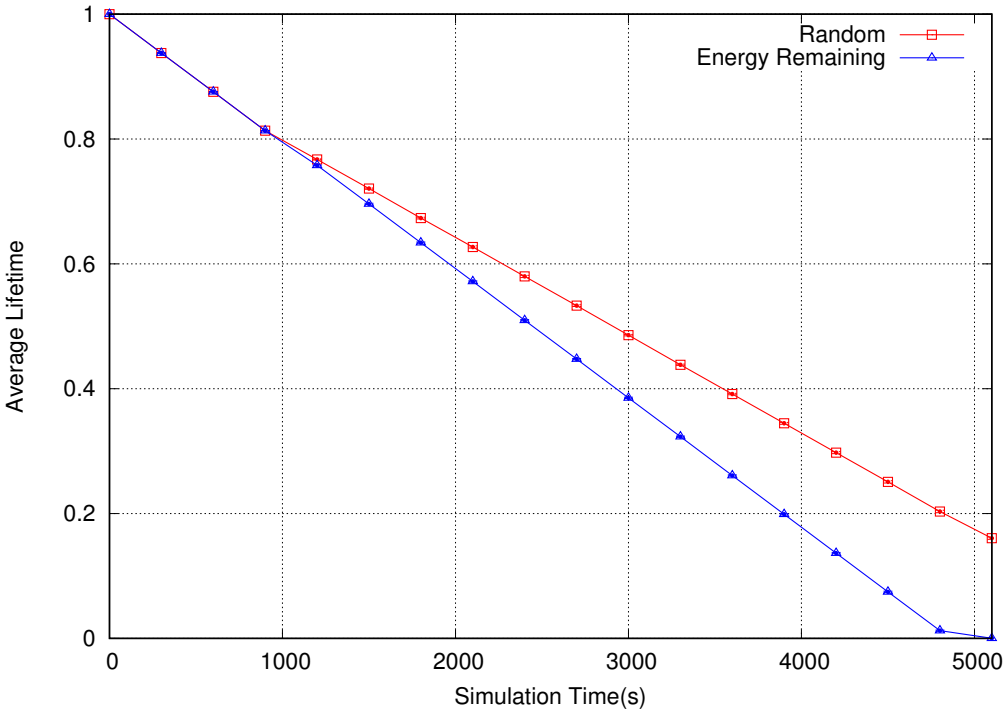

Figure 4.9: Network Lifetime using random numbers and residual energy as rank.

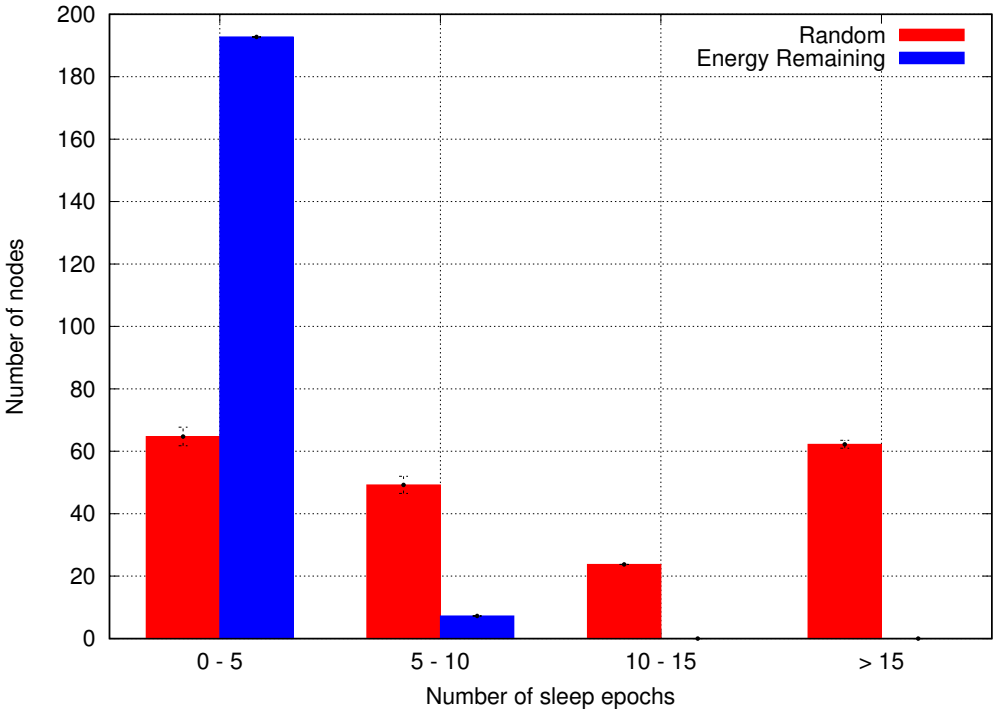

Figure 4.10: Number of sleep epochs using random numbers and residual energy as rank.

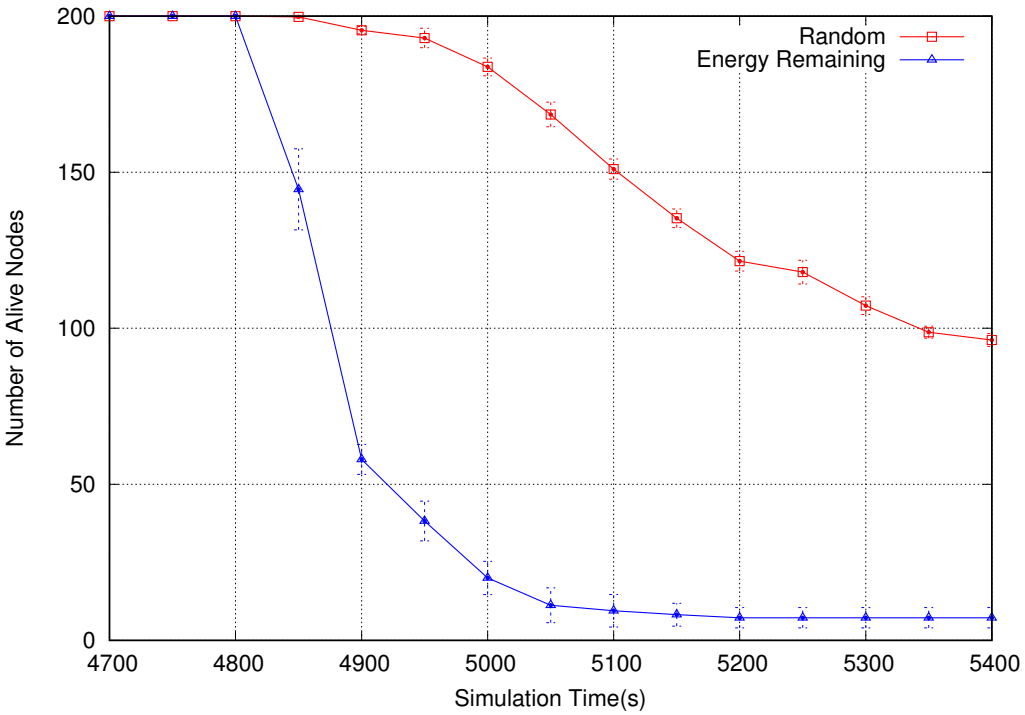

Figure 4.11: Number of alive nodes using random numbers and residual energy as rank.

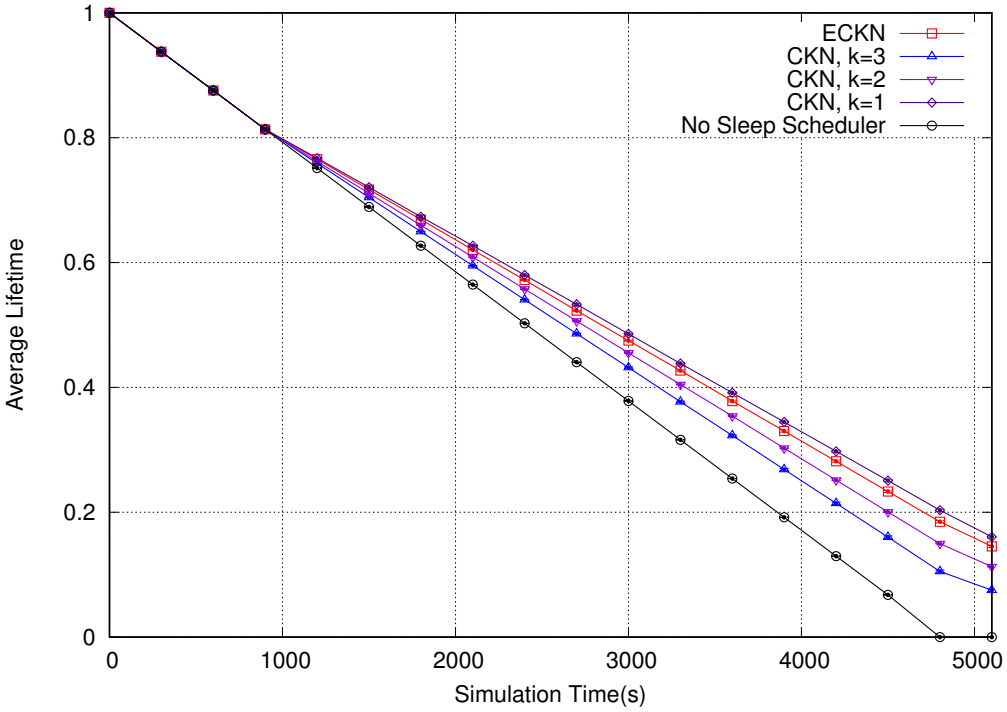

Figure 4.12: Network Lifetime using different sleep schedulers.

of Algorithm 3.1. When I use the residual energy, the node that contains the largest number in a region will never go to sleep due to the first condition. Since the node has the largest rank, all neighbours are going to be considered, if two neighbours are in extremely opposite sides, there will not be a 2-hop connection between them, and this sensor node will not go to sleep. As the node did not sleep, its rank will increase, and the same problem will occur until the node dies. The same problem happens in different regions of the network causing the network lifetime to be worse when using residual energy as rank than when using random numbers. Figure 4.10 shows how many sleep epochs the sensor nodes go to sleep. Results only emphasize what I have mentioned earlier, most of the nodes are sleeping only in 0 to 5 epochs when using the energy remaining as rank. When using random numbers as rank, I have more than 60 nodes having more than 15 epochs where they are sleeping. This explain the better lifetime when using this approach as rank.

Figure 4.11 shows the number of alive nodes at the end of the simulations. As expected, the approach using random numbers as rank for the CKN algorithms presents more alive nodes than the approach with residual energy. These results are the consequence of the problem depicted above regarding the residual energy.

According to my previous results, using random numbers instead of residual capacity of the nodes showed better performance in the network lifetime, therefore I am going to use this approach for my next evaluation. I now assess my ECKN algorithm, comparing its performance to CKN and with a solution using no sleep scheduler. Figure 4.12 shows the network lifetime using the different sleep schedulers. As can be seen, CKN using $k = 1$ presents the best performance, followed by my ECKN algorithm, CKN using $k = 2$, CKN using $k = 3$, and the approach with no sleep scheduler presents the worst performance. These results are explained because CKN using $k = 1$ has the most number of nodes sleeping per epoch, once each node must

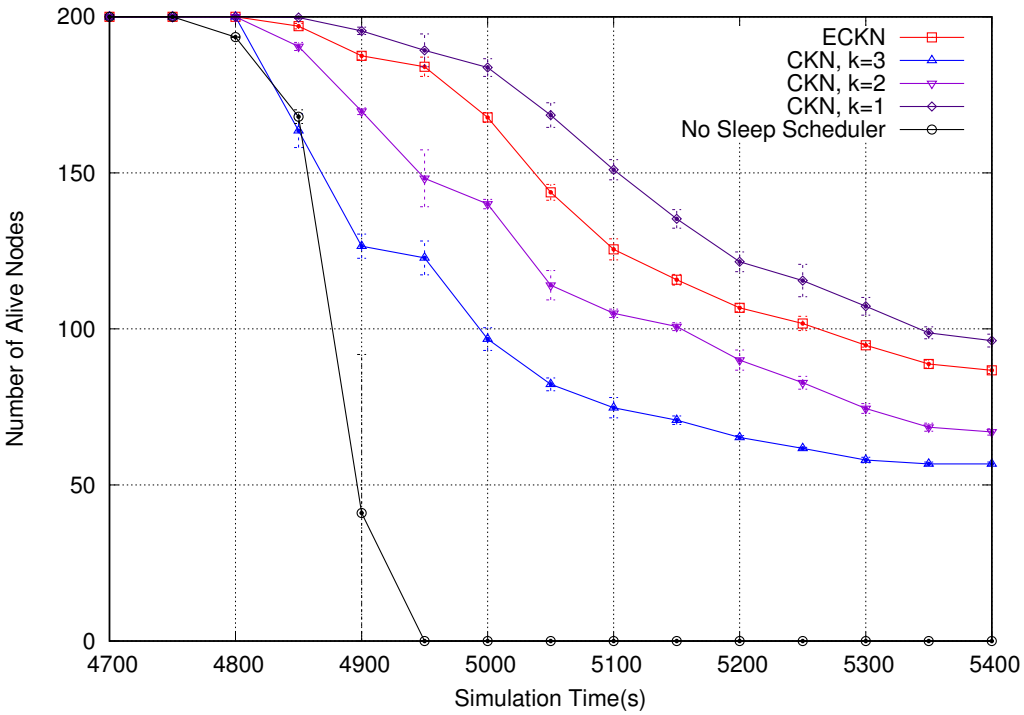

Figure 4.13: Number of alive nodes using different sleep schedulers.

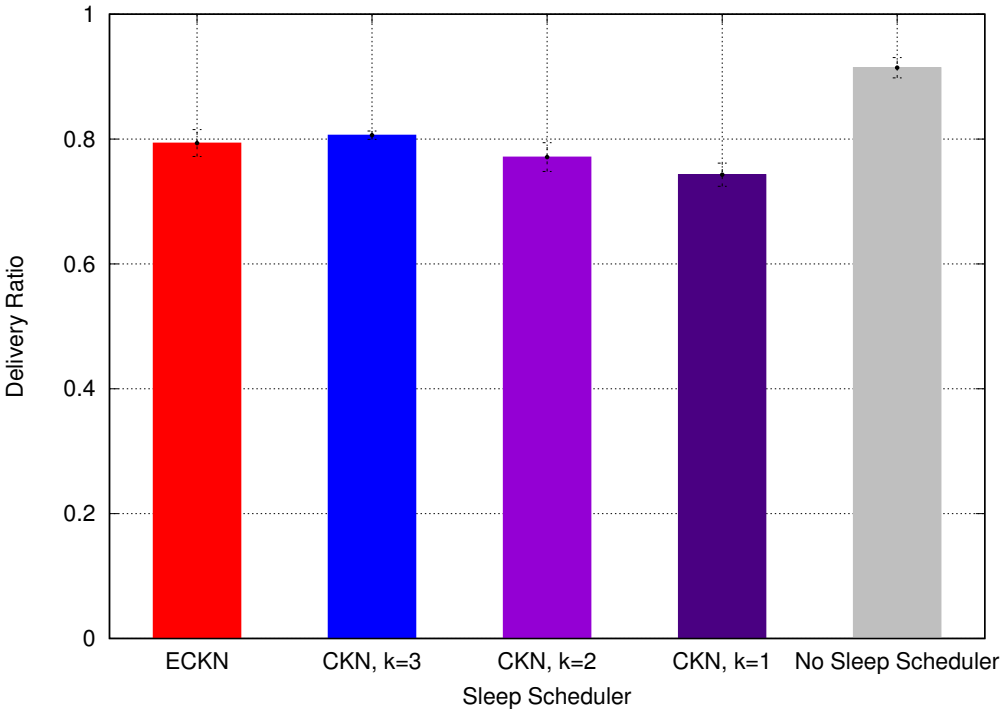

Figure 4.14: Delivery Ratio using different sleep schedulers.

have only one awake neighbour. ECKN is the next because most of the network nodes have $k = 1$, only those closer to the sink have a larger value for k. CKN using $k = 2$ and $k = 3$ perform below because they must have at lest two and three awake neighbours in each epoch, respectively. No sleep scheduler presents the worst performance because no node goes to sleep. Therefore the whole network is awake, decreasing the network lifetime. Figure 4.13 shows the number of alive nodes at the end of the simulations. Following the network lifetime, CKN using $k = 1$ has the best performance, followed by ECKN, CKN using $k = 2$, CKN using $k = 3$, and the approach with no sleep scheduler. Lastly, I evaluate the delivery ratio of the proposed sleep scheduler. As can be seen in Figure 4.14, the approach with no sleep scheduler presents the best delivery ratio, this is due to the fact that no node is sleeping, therefore the best path and shortest path can be used, lowering the probability of dropping a packet. CKN using $k = 3$ shows the second best delivery ratio, following the same idea, when I use $k = 3$, each node must have at least 3 awake neighbours, this increases the probability of finding the best node to forward the packet. Following the same idea, CKN using $k = 2$ presents worse delivery ratio than CKN using $k = 3$ and CKN using $k = 1$ has the worst delivery ratio. ECKN presents similar performance to CKN using $k = 3$, this is due the fact that nodes closer to the sink have larger values for k, increasing the probability of reaching the sink. Therefore, due to the different values of k according to the distance to the sink, ECKN performs as well as CKN using $k = 1$ when I consider network lifetime, while it maintains good delivery ratio, as good as the one presented by CKN using $k = 3$.

Chapter 5

Conclusions

This master thesis proposed ECKN, a novel joint localization estimation, packet routing, and sleep scheduling intended to improve the network lifetime while maintaining acceptable packet delivery ratio in WSNs. The proposed protocol exploits the movement of a mobile sink to estimate each sensor node position without the help of a GPS module. The GPS-equipped mobile sink that shares its position within the network through position packets. Sensor nodes then perform trilateration using the received position packets. The proposed scheme minimizes trilateration problems when using a mobile sink. I have presented two algorithms to find the best reference points: Distance and Closeness. Distance considers the distance between a singular intersection point and every other intersection point calculated. Closeness creates sets of intersection points and checks which set has the closest points. I also presented algorithms to decide whether to accept or not a position packet. As a mobile sink moves, it may send position packets that are not suitable for trilateration. In order to overcome this issue, I proposed Distance Position, Circle Limit and a Hybrid algorithm. Distance Position checks if two position packets have a distance greater than a threshold, and it only accepts position packets that follow this rule. Circle Limit checks if the posi-

tion packet source is within a previous validated position packet circle. If so, it drops the packet. Hybrid takes advantage of both algorithms: It starts by using Distance Position until it gets qualified nodes, then it runs Circle Limit in order to have more accurate estimations over time.

Afterwards, sensor nodes are able to forward event packets to the sink following an algorithm that considers not only the distance to the sink as GGF does, but also the residual energy in each neighbour. In addition, the concept of bridges are introduced, in which the main objective is to shorten paths between the current sink position and its previous positions, thus, reducing the number of intermediate nodes between the source sensor node and the sink.

In order to improve the network lifetime, I proposed a sleep scheduling algorithm based on CKN. CKN is an algorithm intended to maintain the network connected with each node having at least k awake neighbours. My proposed algorithms considers how far the sensor node is from the sink in order to assign a k value to it. Nodes closer to the sink have a larger k value because they are going to receive packets from the whole network in order to deliver it to the sink. Nodes farther from the sink have a smaller k value because the network traffic is not that high in those regions.

5.1 Sensor Node Position Estimation Accuracy

According to my simulation results, the proposed localization schemes have shown acceptable performance. Regarding Reference Points, Closeness outperforms Distance because it obtains the closest set of intersection points considering all possible combinations. Regarding position packet validation, Position Distance shows better performance for applications that demand quick estimations, while Circle Limit shows better performance for applications that demand more accurate estimations.

This is due to the probability of accepting a position packet being higher in Position Distance. However, Circle Limit guarantees that a quadrant will only have a single position packet, which results in better estimations.

5.2 Packet Routing

Regarding the packet routing, ECK with bridges shows the best performance regarding the delivery ratio. It also delivers the packet in a smaller number of hops. This is due the fact that, by creating the bridges, the path between the source node and the sink is shorten, increasing the probability of reaching the sink. However, ECK with bridges also increases the network overhead, since the sensor nodes have to forward the sink position to a farther region. When I remove the bridge algorithm, GGF performs better than ECKN because it considers only the closest neighbour to the sink instead of considering the residual capacity as well. This way, GGF always forward the packet to the best neighbour, however it decreases the network lifetime because it does not take into consideration the power in the nodes.

5.3 Network Lifetime

My proposed sleep schedule approach also showed good performance results. By having different values of k depending on the distance to the sink, ECKN showed similar performance with CKN having $k = 1$ when the network lifetime was analyzed. It also presented equivalent delivery ratio to CKN having $k = 3$. Thus, I can conclude that ECKN saves as much energy as CKN with the most number of sleeping nodes, while it maintains the same delivery ratio as CKN with the most number of awake nodes.

5.4 Future Work

For future work, I plan to investigate the impact of different number of sinks and sink mobility in the position estimation of the nodes and packet routing. I also plan to work on a failure recovery algorithm in the case a packet gets dropped in the packet routing process.

Bibliography

[1] Akkaya, K., and Younis, M. A survey on routing protocols for wireless sensor networks. *Ad hoc Networks 3*, 3 (2005), 325–349.

[2] Akyildiz, I. F., Su, W., Sankarasubramaniam, Y., and Cayirci, E. A survey on sensor networks. *IEEE Communications Magazine 40*, 8 (2002), 102–114.

[3] Akyildiz, I. F., Su, W., Sankarasubramaniam, Y., and Cayirci, E. Wireless sensor networks: A survey. *Computer Networks 38*, 4 (2002), 393–422.

[4] Al-Karaki, J. N., and Kamal, A. E. Routing techniques in wireless sensor networks: A survey. *IEEE Wireless Communications 11*, 6 (2004), 6–28.

[5] Amundson, I., and Koutsoukos, X. A survey on localization for mobile wireless sensor networks. *Mobile Entity Localization and Tracking in GPS-Less Environnments* (2009), 235–254.

[6] Anastasi, G., Conti, M., Di Francesco, M., and Passarella, A. Energy conservation in wireless sensor networks: A survey. *Ad hoc Networks 7*, 3 (2009), 537–568.

[7] Arampatzis, T., Lygeros, J., and Manesis, S. A survey of applications of wireless sensors and wireless sensor networks. In *Intelligent Control, 2005. Pro-*

ceedings of the 2005 IEEE International Symposium on, Mediterrean Conference on Control and Automation (2005), IEEE, pp. 719–724.

[8] Bertanha, M., and Pazzi, R. W. LOGR: Joint localization and geographic routing-based data dissemination in wireless sensor networks with mobile sinks. In *Proceedings of the 6th ACM Symposium on Development and Analysis of Intelligent Vehicular Networks and Applications* (2016), ACM, pp. 83–90.

[9] Bertanha, M., and Pazzi, R. W. JLPR: Joint range-based localization using trilateration and packet routing in wireless sensor networks with mobile sinks. In *2017 IEEE Symposium on Computers and Communications (ISCC) (ISCC 2017)* (2017), pp. 646–651.

[10] Bhuiyan, M. Z. A., Wang, G., and Vasilakos, A. V. Local area prediction-based mobile target tracking in wireless sensor networks. *IEEE Transactions on Computers 64*, 7 (2015), 1968–1982.

[11] Bose, P., Morin, P., Stojmenović, I., and Urrutia, J. Routing with guaranteed delivery in ad hoc wireless networks. *Wireless Networks 7*, 6 (2001), 609–616.

[12] Boukerche, A., Oliveira, H. A., Nakamura, E. F., and Loureiro, A. A. Localization systems for wireless sensor networks. *IEEE Wireless Communications 14*, 6 (2007).

[13] Bulusu, N., Heidemann, J., and Estrin, D. GPS-less low-cost outdoor localization for very small devices. *IEEE Personal Communications 7*, 5 (2000), 28–34.

[14] Cerpa, A., and Estrin, D. ASCENT: Adaptive self-configuring sensor networks topologies. *IEEE Transactions on Mobile Computing 3*, 3 (2004), 272–285.

[15] Chen, B., Jamieson, K., Balakrishnan, H., and Morris, R. Span: An energy-efficient coordination algorithm for topology maintenance in ad hoc wireless networks. *Wireless Networks 8*, 5 (2002), 481–494.

[16] Chen, C.-C., Chang, C.-Y., and Li, Y.-N. Range-free localization scheme in wireless sensor networks based on bilateration. *International Journal of Distributed Sensor Networks 2013* (2013).

[17] Clausen, T., and Jacquet, P. Optimized link state routing protocol (OLSR). Tech. rep., 2003.

[18] Committee, I. C. S. L. M. S., et al. Wireless LAN medium access control (MAC) and physical layer (PHY) specifications. *IEEE Standard 802.11-1997* (1997).

[19] Cota-Ruiz, J., Rivas-Perea, P., Sifuentes, E., and Gonzalez-Landaeta, R. A recursive shortest path routing algorithm with application for wireless sensor network localization. *IEEE Sensors Journal 16*, 11 (2016), 4631–4637.

[20] Erceg, V., Greenstein, L. J., Tjandra, S. Y., Parkoff, S. R., Gupta, A., Kulic, B., Julius, A. A., and Bianchi, R. An empirically based path loss model for wireless channels in suburban environments. *IEEE Journal on selected areas in communications 17*, 7 (1999), 1205–1211.

[21] Falcon, R., Liu, H., Nayak, A., and Stojmenovic, I. Controlled straight mobility and energy-aware routing in robotic wireless sensor networks. In *Distributed Computing in Sensor Systems (DCOSS), 2012 IEEE 8th International Conference on* (2012), IEEE, pp. 150–157.

[22] Forbes, C., Evans, M., Hastings, N., and Peacock, B. Student's t distribution. *Statistical Distributions, Fourth Edition* (2011), 183–186.

[23] Frey, H., and Stojmenovic, I. On delivery guarantees of face and combined greedy-face routing in ad hoc and sensor networks. In *Proceedings of the 12th Annual International Conference on Mobile Computing and Networking* (2006), ACM, pp. 390–401.

[24] Gabriel, K. R., and Sokal, R. R. A new statistical approach to geographic variation analysis. *Systematic Biology 18*, 3 (1969), 259–278.

[25] Godfrey, P., and Ratajczak, D. Naps: Scalable, robust topology management in wireless ad hoc networks. In *Proceedings of the 3rd International Symposium on Information Processing in Sensor Networks* (2004), ACM, pp. 443–451.

[26] Goyal, D., and Tripathy, M. R. Routing protocols in wireless sensor networks: A survey. In *Advanced Computing & Communication Technologies (ACCT), 2012 Second International Conference on* (2012), IEEE, pp. 474–480.

[27] Han, G., Xu, H., Duong, T. Q., Jiang, J., and Hara, T. Localization algorithms of wireless sensor networks: A survey. *Telecommunication Systems* (2013), 1–18.

[28] Han, K., Luo, J., Liu, Y., and Vasilakos, A. V. Algorithm design for data communications in duty-cycled wireless sensor networks: A survey. *IEEE Communications Magazine 51*, 7 (2013), 107–113.

[29] Hofmann-Wellenhof, B., Lichtenegger, H., and Collins, J. *Global Positioning System: Theory and Practice*. Springer Science & Business Media, 2012.

[30] Jiang, M. Cluster based routing protocol (CBRP). *Mobile Ad Hoc Networking Working Group of the Internet Engineering Task Force (IETF)* (1999).

[31] Karp, B., and Kung, H.-T. GPSR: Greedy perimeter stateless routing for wireless networks. In *Proceedings of the 6th Annual International Conference on Mobile Computing and Networking* (2000), ACM, pp. 243–254.

[32] Kirci, P., and Chaouchi, H. Recursive and ad hoc routing based localization in wireless sensor networks. *Computer Standards & Interfaces 44* (2016), 258–263.

[33] Kranakis, E., Singh, H., and Urrutia, J. Compass routing on geometric networks. In *Proceedings of the 11th Canadian Conference on Computational Geometry* (1999).

[34] Langendoen, K., and Reijers, N. Distributed localization in wireless sensor networks: A quantitative comparison. *Computer Networks 43*, 4 (2003), 499–518.

[35] Li, M., Li, Z., and Vasilakos, A. V. A survey on topology control in wireless sensor networks: Taxonomy, comparative study, and open issues. *Proceedings of the IEEE 101*, 12 (2013), 2538–2557.

[36] Mineno, H., Soga, K., Takenaka, T., Terashima, Y., and Mizuno, T. Integrated protocol for optimized link state routing and localization: OLSR-L. *Simulation Modelling Practice and Theory 19*, 8 (2011), 1711–1722.

[37] Nath, S., and Gibbons, P. B. Communicating via fireflies: Geographic routing on duty-cycled sensors. In *Information Processing in Sensor Networks, 2007. IPSN 2007. 6th International Symposium on* (2007), IEEE, pp. 440–449.

[38] Niculescu, D., and Nath, B. Ad hoc positioning system (APS) using AOA. In *INFOCOM 2003. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications. IEEE Societies* (2003), vol. 3, Ieee, pp. 1734–1743.

[39] Oliva, G., Panzieri, S., Pascucci, F., and Setola, R. Simultaneous localization and routing in sensor networks using shadow edges. *IFAC Proceedings Volumes 46*, 10 (2013), 199–204.

[40] Pandey, S., and Varma, S. A range based localization system in multihop wireless sensor networks: A distributed cooperative approach. *Wireless Personal Communications 86*, 2 (2016), 615–634.

[41] Rault, T., Bouabdallah, A., and Challal, Y. Energy efficiency in wireless sensor networks: A top-down survey. *Computer Networks 67* (2014), 104–122.

[42] Reich, J., Misra, V., Rubenstein, D., and Zussman, G. Connectivity maintenance in mobile wireless networks via constrained mobility. *IEEE Journal on Selected Areas in Communications 30*, 5 (2012), 935–950.

[43] Schurgers, C., Tsiatsis, V., and Srivastava, M. B. STEM: Topology management for energy efficient sensor networks. In *Aerospace Conference Proceedings, 2002. IEEE* (2002), vol. 3, IEEE, pp. 3–3.

[44] Sun, Y., Gurewitz, O., and Johnson, D. B. RI-MAC: A receiver-initiated asynchronous duty cycle MAC protocol for dynamic traffic loads in wireless sensor networks. In *Proceedings of the 6th ACM Conference on Embedded Network Sensor Systems* (2008), ACM, pp. 1–14.

[45] Takenaka, T., Mineno, H., Tokunaga, Y., Miyauchi, N., and Mizuno, T. Performance analysis of optimized link state routing-based localization. *IPSJ Digital Courier 3* (2007), 541–554.

[46] Toussaint, G. T. The relative neighbourhood graph of a finite planar set. *Pattern Recognition 12*, 4 (1980), 261–268.

[47] Williams, S. D., Bock, Y., Fang, P., Jamason, P., Nikolaidis, R. M., Prawirodirdjo, L., Miller, M., and Johnson, D. J. Error analysis of continuous GPS position time series. *Journal of Geophysical Research: Solid Earth 109*, B3 (2004).

[48] Wu, G., Wang, S., Wang, B., Dong, Y., and Yan, S. A novel range-free localization based on regulated neighborhood distance for wireless ad hoc and sensor networks. *Computer Networks 56*, 16 (2012), 3581–3593.

[49] Yang, X., and Vaidya, N. H. A wakeup scheme for sensor networks: Achieving balance between energy saving and end-to-end delay. In *Real-Time and Embedded Technology and Applications Symposium, 2004. Proceedings. RTAS 2004. 10th IEEE* (2004), IEEE, pp. 19–26.

[50] Yick, J., Mukherjee, B., and Ghosal, D. Wireless sensor network survey. *Computer Networks 52*, 12 (2008), 2292–2330.

[51] Yuan, Z., Wang, L., Shu, L., Hara, T., and Qin, Z. A balanced energy consumption sleep scheduling algorithm in wireless sensor networks. In *Wireless Communications and Mobile Computing Conference (IWCMC), 2011 7th International* (2011), IEEE, pp. 831–835.

[52] Zhu, C., Yang, L. T., Shu, L., Leung, V. C., Rodrigues, J. J., and Wang, L. Sleep scheduling for geographic routing in duty-cycled mobile sensor networks. *IEEE Transactions on Industrial Electronics 61*, 11 (2014), 6346–6355.

[53] Zigbee, A. Zigbee specification. *ZigBee Document 053474r13* (2006).